(2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California

AD-A242 078

THESIS

INTERNETWORKING WITH
INTERNET PROTOCOL (IP) AND
TRANSMISSION CONTROL PROTOCOL (TCP)
WITHIN THE MILITARY

by

Bruce R. Eikenberg

December, 1990

Thesis Advisor: G. M. Lundy

Approved for public release; distribution is unlimited.

91-14319

91 10 28 098

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: Computer Science Dept. Naval Postgraduate School; 6b. OFFICE SYMBOL (if applicable): 52	7a. NAME OF MONITORING ORGANIZATION: Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code): Monterey, CA 93943-5000	7b. ADDRESS (City, State, and ZIP Code): Monterey, CA 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION; 8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Internetworking With Internet Protocol (IP) and Transmission Control Protocol (TCP) Within the Military

12. PERSONAL AUTHOR(S)
Eikenberg, Bruce R.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Master's Thesis	FROM _____ TO _____	December 1990	114

16. SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Internetworking, Internet Protocol, Transmission Control Protocol

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The backbone of the internetworking technology widely used by the military, as well as many civilian installations, is commonly referred to as TCP/IP. Transmission Control Protocol (TCP) and Internet Protocol (IP) are the two standard communication protocols from which TCP/IP receives its name. By utilizing TCP/IP, the majority of technical issues of interconnecting various computer technologies have become transparent to the user. This thesis conducts an in depth study of many aspects of the TCP/IP technology. Based upon descriptions provided, flowcharts detailing the series of procedures of numerous functions of both TCP and IP are created. Additionally, inefficient TCP/IP functions are discussed and possible solutions to the inefficiencies are provided.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: [X] UNCLASSIFIED/UNLIMITED [ ] SAME AS RPT. [ ] DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL: Lundy, G. M.	22b. TELEPHONE (Include Area Code): (408) 646-2094; 22c. OFFICE SYMBOL: 52LN

Approved for public release; distribution is unlimited.

INTERNETWORKING WITH INTERNET PROTOCOL (IP) AND TRANSMISSION CONTROL PROTOCOL (TCP) WITHIN THE MILITARY

by

Bruce R. Eikenberg
Captain, United States Marine Corps.
B.S., United States Naval Academy, 1983

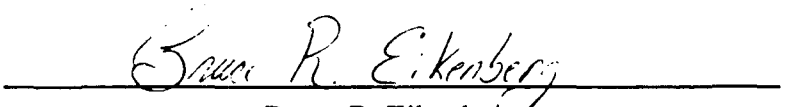
Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
December 1990

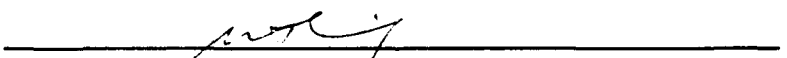
Author:



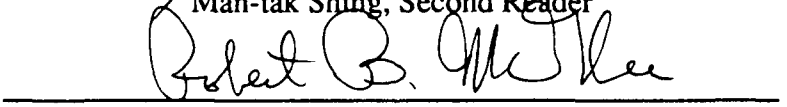
Bruce R. Eikenberg

Approved By:

G. M. Lundy, Thesis Advisor



Man-tak Shing, Second Reader



Robert B. McGhee, Chairman,
Department of Computer Science

ABSTRACT

The backbone of the internetworking technology widely used by the military, as well as many civilian installations, is commonly referred to as TCP/IP. Transmission Control Protocol (TCP) and Internet Protocol (IP) are the two standard communication protocols from which TCP/IP receives its name. By utilizing TCP/IP, the majority of technical issues of interconnecting various computer technologies have become transparent to the user. This thesis conducts an in depth study of many aspects of the TCP/IP technology. Based upon descriptions provided, flowcharts detailing the series of procedures of numerous functions of both TCP and IP are created. Additionally, inefficient TCP/IP functions are discussed and possible solutions to the inefficiencies are provided.

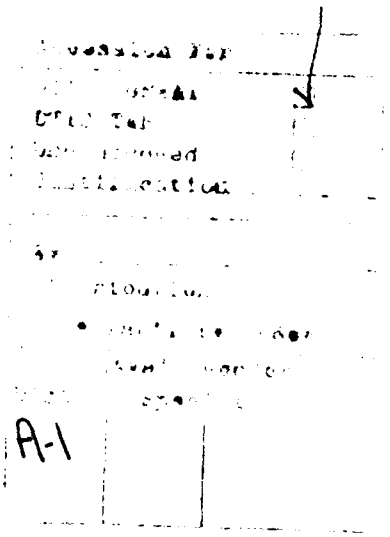


TABLE OF CONTENTS

I. INTRODUCTION 1

A. OBJECTIVES 1

B. RESEARCH QUESTIONS 2

C. ORGANIZATION OF STUDY 2

II. INTERNETWORKING IN THE MILITARY 4

A. BACKGROUND 4

B. OVERVIEW 6

C. FUNDAMENTAL PROCEDURES 7

D. STANDARDS 13

E. SUMMARY 15

III. INTERNET PROTOCOL (IP) 16

A. INTRODUCTION 16

B. FORMAT OF IP DATAGRAMS 20

C. ADDRESSING AND ROUTING 28

1. 32 Bit Internetwork Addressing 28
2. Datagram Routing 29
3. Gateways in the Internetwork 32
4. IP Routing 33
a. Source Host Routing 34
b. Gateway Routing 38
c. Destination Host Routing 41
D. FRAGMENTATION OF DATAGRAMS 43
1. The Need for Fragmentation 43
2. Header Information in Fragmented Datagrams . . . 44
3. Fragmentation Procedure 45
4. Proposed Improvement to the Fragmentation Process 50
5. Comparison of the Existing Fragmentation Procedure and the Proposed Fragmentation Procedure 54
E. REASSEMBLY OF DATAGRAMS 57
F. SUMMARY 65

IV. TRANSMISSION CONTROL PROTOCOL (TCP) 67
A. INTRODUCTION 67

B. FORMAT OF TCP SEGMENTS 69
C. CONNECTION ESTABLISHMENT AND CONNECTION TERMINATION 73
D. RELIABILITY 77
E. FLOW CONTROL 82
F. SUMMARY 86

V. CONCLUSIONS AND RECOMMENDATIONS 89

LIST OF REFERENCES 93

APPENDIX A 95

APPENDIX B 97

APPENDIX C 99

INITIAL DISTRIBUTION LIST 105

LIST OF FIGURES

2.1 Example of an Internet 7

2.2 Comparison of OSI and Military Internetwork That Utilizes TCP/IP 8

2.3 Illustration of Encapsulation of Data 10

2.4 Classes of Networks 12

3.1 Encapsulation of a Datagram in a Network Packet 18

3.2 Format for IP Datagrams 21

3.3 Breakdown of Type of Service Field 23

3.4 Illustration of Flags Field 25

3.5 Extraction of Network Address and Host Identification From a Unique Internetwork Address 30

3.6 Internet Protocol Routing at a Source Host 35

3.7 Computation of a Header Checksum 37

3.8 Internet Protocol Routing at a Gateway 39

3.9 Internet Protocol Routing at Destination Host 42

3.10 Internet Protocol Fragmentation Procedure 47

3.11 Proposed Improvement to the Present Fragmentation Algorithm . 52

3.12 Example of Fragmenting a Datagram Using the Present Algorithm 55

3.13 Example of Fragmenting a Datagram Using the Proposed Modification to the Fragmentation Algorithm 56

3.14 Computation of Total Data Length of Reassembled Datagrams . . 60

3.15 Example of Maximum Reassembly Timer Computation 60

3.16 Reassembly of Fragmented Datagrams 62

4.1 Format for TCP Segments 70

4.2 Source Host Initiates Three Way Handshake 75

4.3 Example of Sequence Number Determination 78

4.4 Flowchart of TCP Within the Source Host 81

4.5 Problem Resulting From a Reduction of the Window Size and Segments Arriving Out of Order 84

4.6 Computation of Permissible Sequence Numbers That can be Sent by the Source TCP 86

I. INTRODUCTION

An expanding field of interest in computer science is that of internetworking. Internetworking is the arrangement of a set of connected networks to form a single powerful network. This is accomplished while individual groups use the network hardware and software best suited for their individual needs.

The internetworking technology currently in use by the military is commonly referred to as TCP/IP. Transmission Control Protocol (TCP) and Internet Protocol (IP) are the two standard communication protocols from which TCP/IP receives its name. Material within this thesis applies to internetworking utilizing TCP/IP.

A. OBJECTIVES

The main objective of this thesis is an in depth study of TCP/IP. The military has used TCP/IP extensively since the Defense Advanced Research Projects Agency (DARPA) dictated the standard in 1983 [Ref. 1].

Additionally, modeling and evaluation of existing TCP/IP technologies will be attempted. Possible improvements concerning the effectiveness and efficiency of existing TCP/IP technologies will also be modeled and evaluated.

B. RESEARCH QUESTIONS

TCP and IP are the two main aspects for internetworking within the military. The concepts of each communication protocol standard will be explored. The details of various procedures within each protocol will be examined to determine the impact on the effectiveness and efficiency of the composite system.

Additionally, congestion within the military internet has been increasing at an alarming rate. The possibility of improving existing TCP/IP technologies to increase efficiency and thus reduce the ever increasing congestion problem will be discussed.

C. ORGANIZATION OF STUDY

Chapter II lays the foundation for the concepts of internetworking in the military. The requirement for a standardization of communication protocols and an overview of internetworking procedures are discussed. Readers unfamiliar with internetworking within the military will find this chapter necessary.

Chapter III provides an in depth study of IP. Detailed information concerning the function of interconnected networks are provided. Datagrams, the basic unit of information passed within IP, are discussed. Specifically, the format of datagrams, addressing, routing, fragmentation, and reassembly of

datagrams are detailed. Models of existing procedures and a suggestion to increase the system efficiency during the fragmentation process highlight the chapter.

Chapter IV provides an in depth analysis of TCP and its functions. The basic unit of data transfer in TCP, the segment is discussed. Information concerning the internal format of segments, the establishment and termination of connections in TCP, the reliability of TCP, and the flow control within TCP is also highlighted.

Chapter V covers the conclusion and recommendations concerning this thesis. Upon completion of this thesis, the reader should have a comprehensible framework for the functions and usefulness of TCP/IP.

Appendix A is a list of acronyms used within this thesis. Appendix B provides a partial list of many wide area networks that are accessible through TCP/IP. Finally, Appendix C diagrams the major computer networks of the Computer Science Department of the Naval Postgraduate School and their relationship to the campus network and the military's interconnected network called Milnet.

II. INTERNETWORKING IN THE MILITARY

A. BACKGROUND

In recent years, the military has experienced a rapid proliferation of strategic and communication networks. Unfortunately for the military, it was found to be impossible to construct one single network that would be universally acceptable to all users. Most military networks are independent and are designed to accomplish the goals of an individual group. The ultimate goal of internetworking is to hide the technical differences between networks and make communication between the networks independent of all underlying hardware technology.

The competitive procurement process within the Federal Government has complicated the problem of communication between networks. The competitive process awards the purchase of the desired network to the lowest bidder who can accomplish the stated objectives at the individual installation. This has resulted in a variety of network technologies and a multitude of data processing equipment from various vendors.

This variety of equipment and software is a problem because hardware and software developed by one vendor is not always directly compatible with the equipment and software of other manufacturers. While the procurement

of an assortment of networks continued, the military discovered a need to exchange data files and messages. It also found that it would be extremely cost effective to share expensive resources such as high powered control processing units, printers, and mass storage devices. In order to share this information with the multitude of various equipment and users, a standard communication protocol was necessary to support the interconnection of these various networks.

Internet is a set of connected networks that act as one powerful network. This is accomplished while individual groups use the network hardware and software best suited for their individual needs.

Thousands of users located at a multitude of locations depend on internetworking for their daily activities. The main Department of Defense applications of internet include:

Simple Mail Transfer Protocol (SMTP) - electronic mail.
File Transfer Protocol (FTP) - used for file transfers.
Telnet - remote login capabilities.

SMTP is the most useful and most frequently used service of the military internetwork. It permits users to send messages electronically to one another. Messages from other host computers are recognized as incoming mail and translated to a compatible format that the destination can understand. The File Transfer Protocol (FTP) allows users to transfer files from one

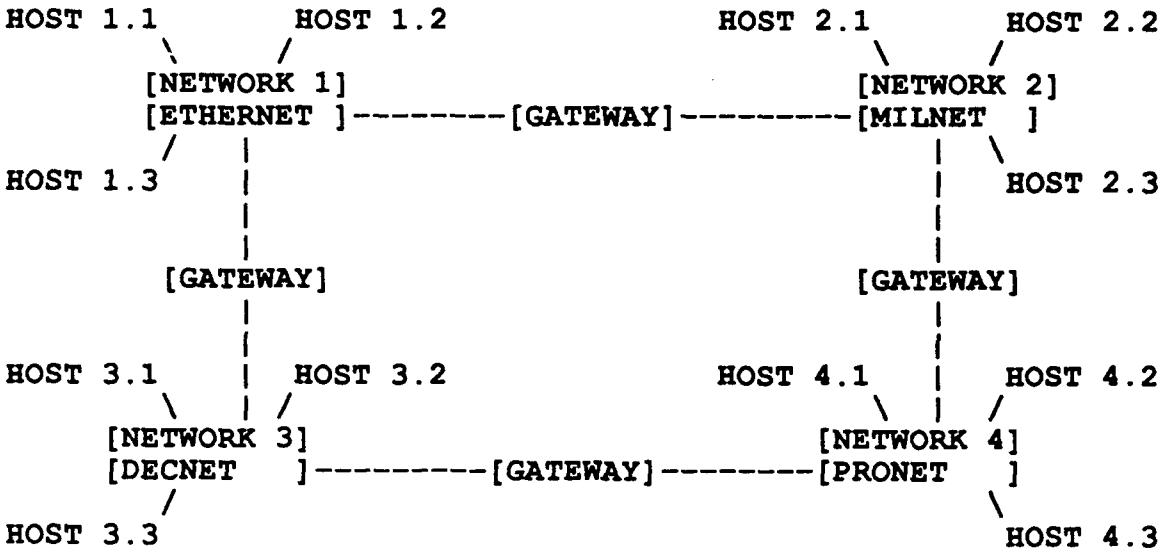
computer to another. This can be accomplished even if each computer has a different operating system or a different file storage format.

Telnet allows the user to establish a connection with a remote host computer from his local host computer. Once the connection has been established and the user has successfully logged in, the user can manipulate the remote computer as if he were logged on directly.

B. OVERVIEW

Internetworking satisfies the communication requirements in the connection of mixed individual networks. A connection between associated individual networks is formed by a gateway. A gateway has the ability to read data from one network and send the data, in an appropriate format, to another network.

Using the internet, any two hosts on any of the connected networks may reliably exchange data. Figure 2.1 depicts an internet of various individual networks connected by gateways. The goal of the internet is to hide the technical differences between networks and enable data transfer despite the various network technologies involved. This communication is accomplished while each individual network maintains the network hardware and software that is optimal for their daily activity.



HOST - is a computer attached to a network.
GATEWAY - is a link between individual networks.

Note - each host may communicate with any other host.

Figure 2.1 Example of an Internet

C. FUNDAMENTAL PROCEDURES

Each host, connected to the internetwork, has the ability to execute certain processes which require the use of Internet communication protocols. These protocols allow the many networks that comprise the internetwork to act as one single reliable network. This is accomplished by the use of universal conventions that specify the details of data transfer.

The military internetwork uses a four layer approach. The four layers used are as follows:

1) Network Layer
2) Internet Layer
3) Transmission Control Layer
4) Process Layer

In contrast, the international standard, known as Open System Interconnection (OSI), uses a seven layer approach. Figure 2.2 [Ref 2] directly compares the seven layer OSI architecture and the four layer military internetwork that uses TCP/IP.

<table>
<tr><th>MILITARY INTERNETWORK WITH TCP/IP</th><th>OSI</th></tr>
<tr><td rowspan="2">PROCESS LAYER</td><td>APPLICATION</td></tr>
<tr><td>PRESENTATION</td></tr>
<tr><td>TRANSMISSION CONTROL LAYER</td><td>SESSION</td></tr>
<tr><td>INTERNET LAYER</td><td>TRANSPORT</td></tr>
<tr><td rowspan="3">NETWORK LAYER</td><td>NETWORK</td></tr>
<tr><td>DATA LINK</td></tr>
<tr><td>PHYSICAL</td></tr>
</table>

Figure 2.2 Comparison of OSI and the Military Internetwork That Utilizes TCP/IP

The network layer accepts data from the Internet layer and adds certain control information to this data, thus making a packet. It is also the means

in which data is exchanged between the host and the network or exchanged between two devices connected to the same network.

The Internet layer accepts information from the Transmission Control layer and adds certain control information to this data. The Internet layer passes or receives the concatenation of this information, called an Internet Protocol datagram, to the Network layer for further processing.

Each host and gateway contains the Internet layer. The Internet layer is responsible for the following:

1) Properly addressing the datagrams.
2) Routing the datagrams.
3) Fragmenting and reassembling datagrams.
4) Simple forms of validation and flow control.

The Transmission Control layer packages portions of information from the Process layer and adds certain control information to this data, thus making a segment. The Transmission Control layer provides reliable end-to-end transport of data between host processes. It ensures that data from the Process layer is delivered and received without errors, in the proper order, and without data loss or duplication.

The Process layer contains the protocols needed to support the various application programs that may be executed within a host. The Process layer communicates with the Transmission Control layer to send or receive data.

Each layer has its own important functions. The encapsulation process is illustrated in Figure 2.3 [Ref 3].

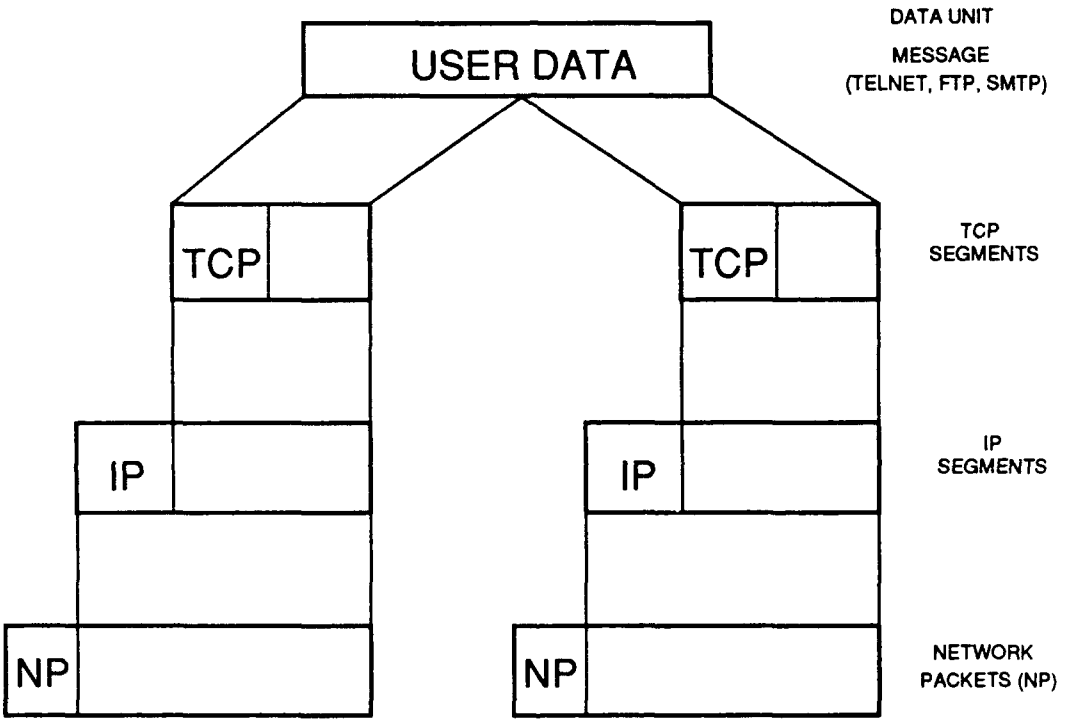


Figure 2.3 Illustration of Encapsulation of Data

Each message has a distinct source address and a distinct destination address. These addresses consist of three important entities as listed below:

1) Network identification
2) Host identification
3) Port identification

Each host on the Internet is assigned a unique 32 bit Internet address. The network identification and host identification are encoded in this Internet address. Efficient routing of information is sustained as the network identification is extracted from this address.

Gateways depend on this extraction to determine if the destination network is directly attached to the gateway or must be routed to another gateway.

There are three classes of networks. The following describe the classes:

Class A: Class A networks can be very large. There may be more than (65,536) hosts connected to this class of network. Very few of these networks are in existence. The MILNET is one example of a Class A network.

Class B: Class B networks are medium sized networks. There can be up to 65,536 hosts connected to Class B networks. Typically, Class B networks have more than 255 hosts and less than 1000 hosts connected.

Class C: Class C networks are relatively common. There are less than 255 hosts connected to these types of networks. Most standard Ethernet networks fall into this category.

As indicated in Figure 2.4 below [Ref 4], the high order bits can immediately determine the network class. Hence, the network identification and host identification may be efficiently extracted.

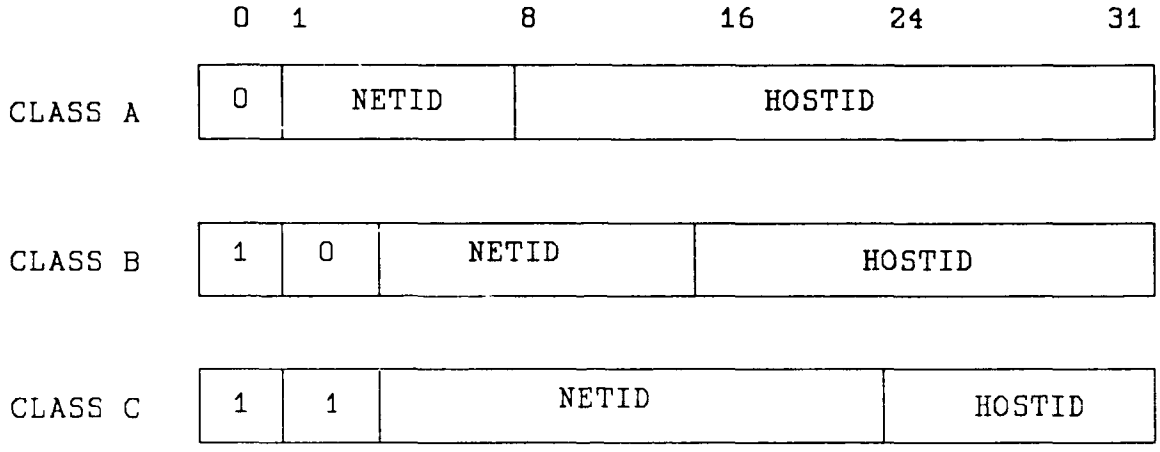


Figure 2.4 Classes of Networks

Multiple users within each host have the capability of simultaneously using the internetwork to exchange data with remote processes. For this reason, multiple processes within each host are further distinguished by ports.

The entire address identifies a socket. Sockets are unique within the Internet. A socket can be compared to a long distance telephone call as indicated below:

AREA CODE	PREFIX	EXTENSION
Network identification	Host identification	Port identification

D. STANDARDS

Internetworking is intended to support various networks, multiple hosts on each network, and a multitude of processes within each host. This is accomplished by implementing standard protocols for all associated networks. The Defense Communication Agency (DCA) implemented the following military standards [Ref 5].

1. MIL-STD-1777 Internet Protocol (IP):
   IP provides a connectionless communication service to transfer data across one or more connected networks. IP does not make any assumptions concerning network reliability.

2. MIL-STD-1778 Transmission Control Protocol (TCP):
   TCP is a transport protocol that ensures reliable, error free, end-to-end transport of data between host processes.

3. MIL-STD-1780 File Transfer Protocol (FTP)

4. MIL-STD-1781 Simple Mail Transfer Protocol (SMTP)

5. MIL-STD-1782 Telnet Protocol

The standards chosen by the DCA were developed and refined as part of the ARPANET research project. These standards allow for interoperability of equipment and simplified procurements. However, numerous disadvantages are associated with DCA's standards. DCA's standards, like most standards, potentially inhibit innovation and the possibility of finding superior solutions. Also, DCA's standards were developed prior to the announcement of the

international standards. Hence, TCP and IP do not conform with the international standards that are widely used in today's market place.

Failure to conform to international standards has further complicated the procurement process. There are a limited number of manufacturers who will continue to support the non- international standard hardware and software used by the military. By limiting the number of manufacturers, competition is decreased and prices tend to increase. Also, international standards are continually evolving and improving. DCA's standards have essentially remained stagnant.

The military is aware of the problems associated with non- standard hardware and software. Commitments have been made to eventually migrate to international standards. According to sources at the DCA, initial migration to international standards will begin as early as 1995 [Ref 6]. A radical change to international standards has been determined to be not feasible because of the abundance of equipment currently in use.

Future procurements are expected to support both DCA's standards and international standards. Eventually, the life cycle of the older equipment will expire and the transition to international standards can be accomplished. As for the present, TCP and IP dominate the internetworking of the Department of Defense.

E. SUMMARY

Within the military, there is a wide variety of seemingly incompatible equipment. Internetworking with TCP and IP enables these individual networks to act as one single powerful network. This is accomplished while the individual networks utilize the network hardware and software that is best suited for their daily activities.

TCP and IP combine their standard communication protocols to form a common basis for internetworking in the military (TCP/IP). Many individuals depend heavily upon the internet to carry on their daily activities.

TCP/IP was developed and implemented prior to the announcement of the international standards for internetworking. Hence, TCP/IP varies from the procedures used in the majority of today's marketplace. Eventually, it is expected that TCP/IP will be replaced or will evolve towards the international standards.

III. INTERNET PROTOCOL (IP)

A. INTRODUCTION

The interconnected networks used within the United States military rely heavily on IP. Recall that the military's internetwork is comprised of a variety of network technologies and a multitude of data processing equipment from various vendors. IP enables all machines attached to this internetwork to share universal rules in transferring data across a variety of equipment and software.

There is an IP module within each host and each gateway attached to the internetwork. Each network attached can remain independent and accomplish the mission of an individual group of users while supporting IP.

The underlying details of the IP are hidden from the user. It provides a connectionless communication service to transfer data across one or more connected networks. IP makes no assumptions concerning the reliability or the technical aspects of any network attached to this internetwork.

IP creates a datagram to transfer data. This datagram consists of data from the Transmission Control layer (which is all or part of the data from the actual application program) concatenated with certain control information. This control information assists in one or a combination of the following:

1. Addressing datagrams properly.

2. Routing datagrams.

3. Fragmentation of datagrams.

4. Reassembly of fragmented datagrams.

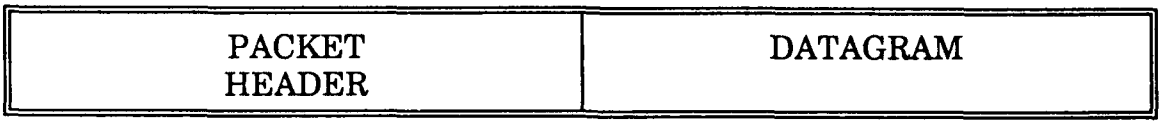
5. Reduction of congestion within the Internet by use of the datagram lifetime timer.

6. Verification of the correctness of datagram's control information.

7. Minimal control of error reporting and flow control using Internet Control Message Protocol (ICMP).

Datagram delivery is provided by IP from the source host, to the destination host, via all subsequent gateways that the internetwork utilizes while the datagram is en route to its final destination. When accepting an incoming datagram, each IP module will either "send" the datagram towards its destination or "deliver" the datagram if the datagram has reached its final destination.

The IP must coordinate with the Network layer. In order for the datagram to be transmitted through a local network, the datagram must be embedded in a local network packet. The packet is the means by which data is transferred between the host and the network or exchanged between two devices connected to the same network. The details of the network packet vary depending upon the individual network in use. Figure 3.1 illustrates a datagram embedded within a packet.



PACKET HEADER - Contains the final destination address or the address of a gateway required in route to the final destination.

DATAGRAM - Data from the Transmission Control Layer concatenated with Internet Protocol Control Information.

Figure 3.1 Encapsulation of a Datagram in a Network Packet

At each subsequent gateway, the datagram is extracted from the network packet and examined to distinguish which subsequent network the datagram should be delivered to next. The datagram control information is modified as necessary and routed from the gateway to its final destination or subsequent gateway towards its final destination.

Networks vary in their capabilities. Specifically, networks may vary in the size of packets that may be passed within a network. Gateways may fragment a datagram if it is necessary for transmission over the next network. Subsequent fragments may be further fragmented, if necessary, to enable transmission of an acceptable sized network packet. The datagram will be reassembled to its original form at its final destination.

Successful delivery of each datagram is quite reliable but not guaranteed. IP does not receive acknowledgments of delivery of data and has no indication that the data was delivered without errors. Reliability of the

transmission of data and the correctness of the actual data is the responsibility of a higher level protocol.

Datagrams may be discarded from the internetwork due to errors discovered in the header information of the datagram, congestion in the network, system failures, or time expiration of the datagram. Eventually, the Transmission Control layer will discover the problem of delivery and retransmit the data via the IP. Chapter IV provides detailed information on the functions of the higher level Transmission Control layer.

Each datagram is treated individually by the network. Hence, routes from the same source host to the same destination host may vary in response to system failure or congestion within the internetwork. Varying routes for datagrams may lead to datagrams being delivered out of order. Once again, it is the responsibility of the Transmission Control Layer to rectify this problem.

This dynamic routing method used by IP gives the system the flexibility to make routing decisions based upon the current network topology and the amount of congestion experienced by the internet at a particular instant in time.

In contrast, static routing would utilized the same path in delivering datagrams despite the amount of congestion that may be encountered. This may lead to datagrams being discarded and hence, inefficiencies in system

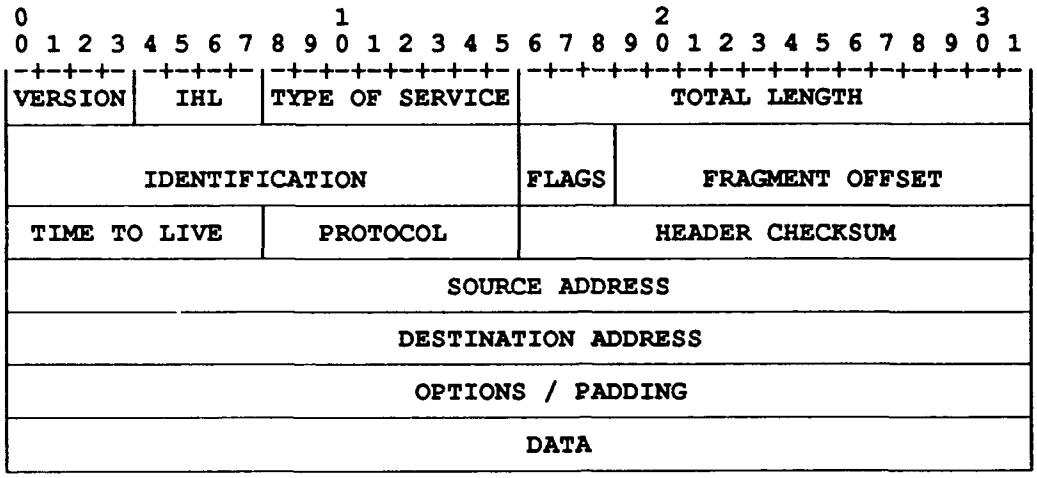
performance would be observed. Dynamic routing is more efficient than static routing because of the possibility of internal system failure and the abundance of additional overhead that is required for the storage of fixed routes.

IP enables the internetwork to share universal rules when transferring data across a variety of network technologies. The underlying details of the IP are hidden from the user and IP makes no assumptions concerning the reliability or the technical aspects of any attached network. The following sections examine the Internet Protocol in detail.

B. FORMAT OF IP DATAGRAMS

Each datagram within the internetwork consists of segments of data from the Transmission Control layer concatenated with control information. This control information is universally understood by all IP modules and is necessary for the successful transmission of datagrams. This section details the specific entities that combine to form the complete datagram.

Within every datagram, there are at least 192 bits of control information that precede the data segment. There is a possibility that more control bits are required depending upon certain options that may be selected by administrative users. In any case, the number of bits preceding the data segment of a datagram will be at least 192 bits and will always be a multiple of 32 bits. Figure 3.2 illustrates the datagram format [Ref 7].



All values are binary representations.

Figure 3.2 Format for IP Datagrams

The first four bits indicate the version. IP was created with the possibility for future improvements and the version reflects the latest protocol currently in use. It is imperative that the sending host, receiving host, and all subsequent gateways that handle the datagram agree that their respective IP module version number and the version number of the datagram are identical. A mix in version numbers could possibly result in erroneous transmission of data. The current version of IP is four. Because of the eventual migration to international standards, this version number is not expected to change.

The next four bits indicate the Internet Header Length (IHL). This field indicates the length of the control information associated with the datagram in units of 32 bit words. Recall that the control information is always a

multiple of 32. The minimum possible number for this field is five. This value may increase variably if options for the datagram are specified.

The following eight bits comprise the Type of Service that the user may specify. These eight bits are further divided into the following fields:

1. Precedence of Data. (3 bits)
2. Delay Indication [normal or low]. (1 bit)
3. Throughput Indication [normal or high]. (1 bit)
4. Reliability Indication [normal of high]. (1 bit)
5. Remaining bits are reserved for future use. (2 bits)

Type of Service is passed from IP module to IP module a the datagram passes through an interconnected set of networks. It is important to note that the Type of Service is not guaranteed.

Unfortunately, because of the independent nature of many of the connected networks, sometimes the desired service is not available. However, the values of the type of service are passed to the next IP module where the service may be provided. Figure 3.3 is an illustration of the breakdown of the Type of Service field.

P - Precedence
D - Delay
T - Throughput
R - Reliability
F - Future use

Figure 3.3 Breakdown of Type of Service Field

The precedence indicates the importance of the datagram. Preferential treatment should ideally be afforded to the datagram with a higher precedence. A normal value is indicated by the binary representation of the zero value. The maximum precedence is indicated by the binary representation of seven.

The delay indication will handle the datagram normally, as indicated by the system performance at the time of processing, if a '0' is indicated. If a '1' is indicated the system will take the path of least delay. For instance, if the datagram had the option of being routed over a dedicated line or a high delay satellite link, and a '1' was indicated, the dedicated line would be selected. If a '0' had been selected, the system would have routed the datagram depending upon system performance and congestion at the time of processing.

Likewise, the throughput indication either indicates to the system to take the path of highest throughput if a '1' is selected or, select the path

independently if a '0' is selected. Typically, satellite links have high throughput.

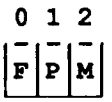
Reliability can also be preferred. Given an option between a connected or connectionless path, and a '1' is selected indicating high reliability, the connected path would always be chosen. Otherwise, if a '0' is selected, the system will route the datagram depending upon system performance and congestion at the time of processing.

The Total Length field contains the total number of bytes of both the control information and the data segment of the datagram. The length of the data can be derived from the difference between the Total Length and a function of the Internet Header Length. Recall that the Internet Header Length is in units of 32 bit words. The following formula derives the length of the data:

Length of Data = Total Length - (Internet Header Length * 4)

The second 32 bit word has a 16 bit field that is used for datagram identification. This identification is uniquely assigned by the source host to each datagram. This identification number, in conjunction with other entities, is necessary for reassembling fragmented datagrams at the destination host.

The Flags field is used for the purposes of fragmenting datagrams. The Flags field is broken down into three bits. Figure 3.4 illustrates the flags field.



F - Future use
P - Permission to fragment
(0 indicates OK to fragment)
(1 indicates DO NOT fragment)
M - More fragment to follow
(0 indicates last fragment)
(1 indicates more fragments)

Figure 3.4 Illustration of Flags Field

The zero bit is unused. The one bit determines if the datagram may be fragmented. If a destination is known to not have the capability to reassemble datagram fragments, then the source will set the permission bit to a one to indicate that this datagram is not to be fragmented. The system will simply discard the datagram if it cannot be delivered to the destination without fragmentation. Once the datagram has been fragmented the last bit of the flags field indicates whether more datagram fragments will follow or whether it is the last fragment of that particular datagram.

The last 13 bits of the second 32 bit word represent the Fragment Offset. This field is used if the datagram has been fragmented. It indicates the placement of this particular fragment when reassembling the original datagram. So the first fragment always has an offset equal to zero.

The Time to Live (TTL) field is used to limit the time that a datagram can remain within the internetwork. A datagram is discarded when the TTL equals zero. Each time that an IP module processes a datagram, the TTL field is decremented. Therefore, it is not a clock but actually a counter. The TTL field reduces congestion by eventually ridding the system of undeliverable datagrams. Because the TTL field contains eight bits, the datagram may be processed a maximum of 255 times by IP modules within the internetwork. The source host determines the initial value for the TTL field.

The Protocol field indicates the next level protocol that will process the received data. For the purposes of this paper, the next level protocol will always be TCP.

The Header Checksum ensures the validity of the header information by dividing the control information of the datagram into 16 bit words, summing the binary values of these words, and then taking the one's complement of their sum. This checksum ensures the integrity of the header information only. The validity of the data segment of the datagram is not guaranteed within IP.

The 32 bit source address and destination address are unique addresses that identify both the network identification and the host identification. A detailed explanation of the internetwork address is given within Section C of this chapter.

Options are available to assist in monitoring and debugging the network. The network can be monitored by enabling the system to list its particular route from source to destination or by time-stamping the datagram, at each IP module, to indicate processing speed. The option of directing a datagram through a particular gateway can be useful in debugging.

Padding simply adds additional binary zeros to the IP header to ensure that the length of the header is a multiple of a 32 bit word. Each datagram will contain padding between zero and 31 zero bits depending upon the options selected. Padding is only used within datagrams when options have been selected.

The DATA simply refers to the data segment associated with each datagram. Because of limitations within the header, the following is the maximum length of the data field:

Maximum Data Length = Maximum Total Length (65535) - (Internet Header Length * 4)

IP shares communication rules in processing datagrams without regards to reliability of the network. It is clear that the control information of the datagram is designed to be universally read by IP modules which must process the datagram prior to delivery to its destination. IP modules continue to recalculate the control information after processing each datagram because the

TTL field is always decremented and the possibility exists that fragmentation of the datagram is necessary.

C. ADDRESSING AND ROUTING

Two of the basic functions of military internet are addressing and routing. Addressing is the identification of the ultimate destination of the datagram. Routing is finding a path for the datagram to reach its destination. IP modules within each host and gateway are responsible for addressing and routing.

1. 32 Bit Internetwork Addressing

Each host connected to the internetwork is assigned a unique 32 bit internetwork address. The host's network address and host identification are encoded within this internetwork address.

By examining the 32 bit internetwork address, IP can easily determine the classification of the network, the destination network address, and the host identification within the destination network. The network classification is distinguished by the zero bit, or by the combination of the zero bit and the one bit, of the 32 bit internet address. Figure 3.5 is an illustration of how efficiently the IP can determine the network classification simply by examining the first few bits of the internetwork address. Once the network

classification has been determined, the network address and host identification can be easily extracted.

2. Datagram Routing

Routing a datagram to its destination can either be accomplished directly, if the destination is on the same network, or indirectly if the destination is on a connected network. For direct routing, the datagram is encapsulated into a network packet and transmitted directly from one machine to another by use of the Network layer. Indirect routing requires a more complex approach. Indirect routing forces the sender to pass a datagram, via the Network layer, to a gateway for delivery.

Routing decisions are based upon routing tables that reside within each host and gateway. Routing tables contain the next node that a datagram should pass through based upon its final destination.

For efficiency in speed of processing and storage, routing tables base their route decisions on the network identification (NID) portion of the destination address. Recall that the NID can be quickly extracted once the class of the destination address has been identified.

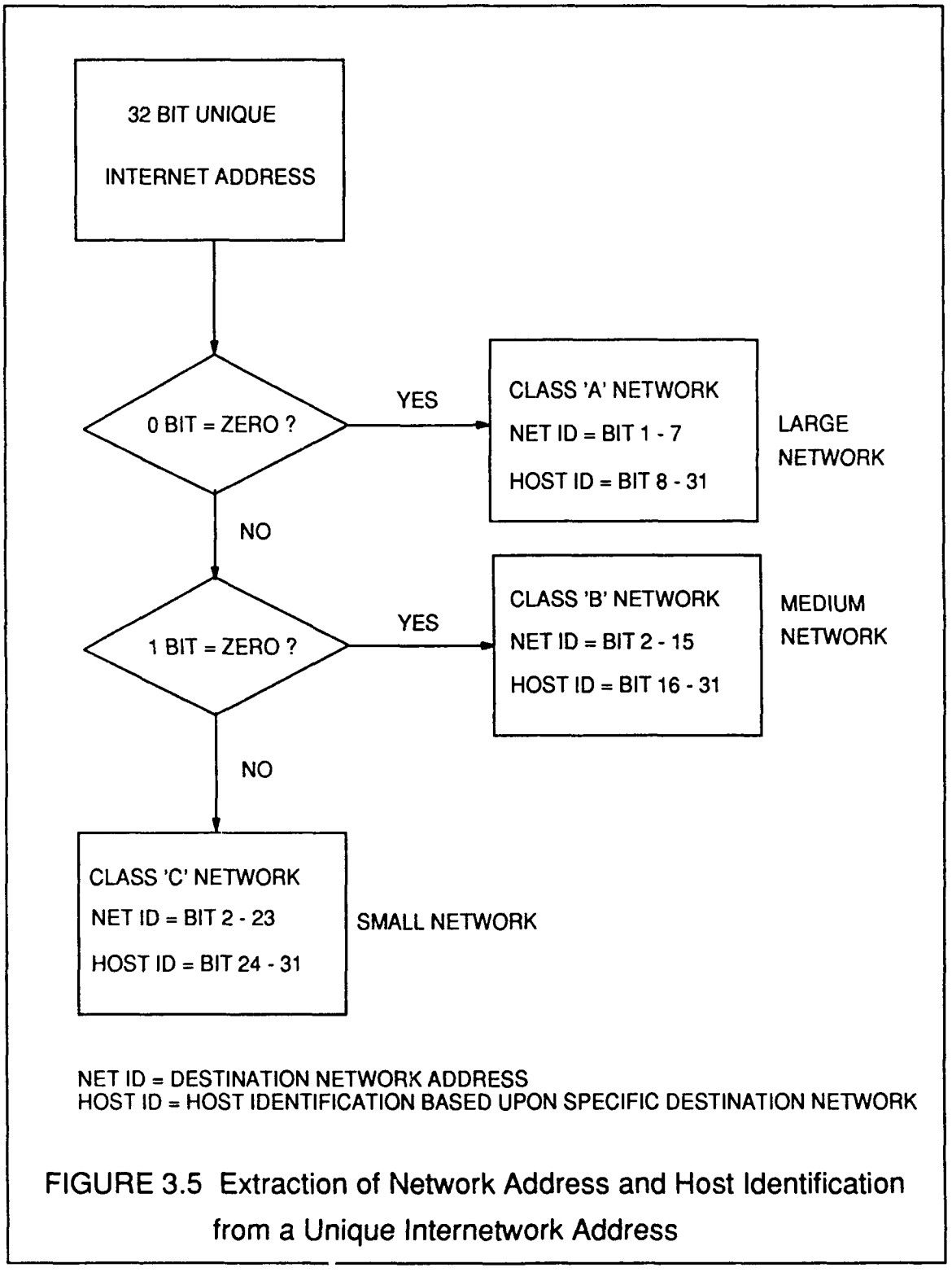


FIGURE 3.5 Extraction of Network Address and Host Identification from a Unique Internetwork Address

Routing tables may be fixed or dynamic. Typically, routing tables are obtained from secondary storage devices at startup. Clearly, dynamic routing tables are more flexible in responding to system failures and congestion than static routing tables because dynamic routing tables can be continually updated to reflect the current system performance.

Routing tables can also support security and priority of a datagram. The routing table can select its next path based, not only upon its final destination, but also as determined by the type of service dictated by TCP.

A complete route table, listing the routes of all possible destination addresses, is impractical for each host and gateway. Such a routing table would have enormous storage and maintenance requirements. Hence, a default value is usually given in the routing table to direct the datagram to a subsequent IP module. The next IP module may have the destination within its route table or, once again, default to another subsequent IP module for processing. This processing continues until either the datagram reaches its destination, the datagram is discarded because of an error in the checksum, or the datagram is discarded because of an expiration in the TTL field.

A difference between the IP module within the host and within the gateway is that a gateway routing table is usually better informed about

the entire internetwork. Typically, a host route table will contain the internetwork addresses of all directly attached hosts and will default to the nearest gateway if delivery cannot be accomplished.

3. Gateways in the Internetwork

Gateways are a vital link in the internetwork. A gateway connects two or more individual networks. In each network that it connects, the gateway is considered to be a connected host with a specific IP address.

Gateways have the ability to accept datagrams from one network and forward the datagram to an attached network. This transition of the datagram is accomplished despite the varying technological differences that may exist between the two connected networks.

Gateways also validate the datagram. They evaluate the datagram's header checksum, and ensure that the TTL field does not equal zero. If problems exist in the header checksum or the TTL field, then the datagrams are discarded.

Connected networks may vary in their capabilities. Specifically, the size of an acceptable network packet may differ. Gateways have the ability to fragment datagrams if necessary. This fragmentation reduces the packet size to a length that can be handled by the next network.

Basically, a gateway decapsulates incoming datagrams, validates the datagram, determines the next IP module required for the datagram's

delivery, fragments the datagram if necessary, and then encapsulates the datagram for transmission over the next network. If the next IP module cannot be determined, an error message is generated.

There are two types of gateways in the internetwork. A small number of the gateways are classified as "core" gateways. Core gateways contain a complete list of all routes to all destinations. No default values are provided and if a route cannot be determined by a core gateway, an error message is generated to indicate that the destination is unavailable or unreachable. Core gateways are centrally controlled and are continually updated to reflect the current status of the entire internetwork. The more common type of gateway contains a partial routing table as well as a default value. The default route is selected if the desired address is not contained in the routing table. Eventually, these gateways will deliver the packet of information to its final destination or to a core gateway for specific routing towards its final destination.

4. IP Routing

IP routing is active in all hosts and gateways. The routing functions are different for the source host, destination host, and all subsequent gateways used in the transmission of a datagram. IP does not guarantee successful delivery of any datagram. It is up to the higher level TCP to determine reliability and delivery of error free data.

The source host, gateway, and destination host utilize the capabilities of the IP module quite differently. Figures 3.6, 3.8, and 3.9 describe the IP module functions for a source host, gateway, and destination host respectively.

a. Source Host Routing

Within a source host, the beginning stages of the IP communication process are initiated. Figure 3.6 illustrates a flow chart of the IP processes that occur within a source host. The numbers in parentheses correspond to the box numbers of Figure 3.6.

A segment of data is passed from the higher TCP to the IP for processing (1). The data segment accepted from the TCP contains useful information that IP must utilize for successful transmission. Specifically, information concerning the source address, destination address, protocol, type of service, length of segment, time to live, and options selected, is provided (2).

Each datagram created at the source has its own unique identification number to uniquely identify the datagram within the internetwork. This identification number is required in reassembling datagrams at the destination should the original datagram become fragmented during transmission (3). Depending upon the options selected and the length of the data segment, IP can determine the value for the total length and the Internet Header Length for the newly created datagram (4).

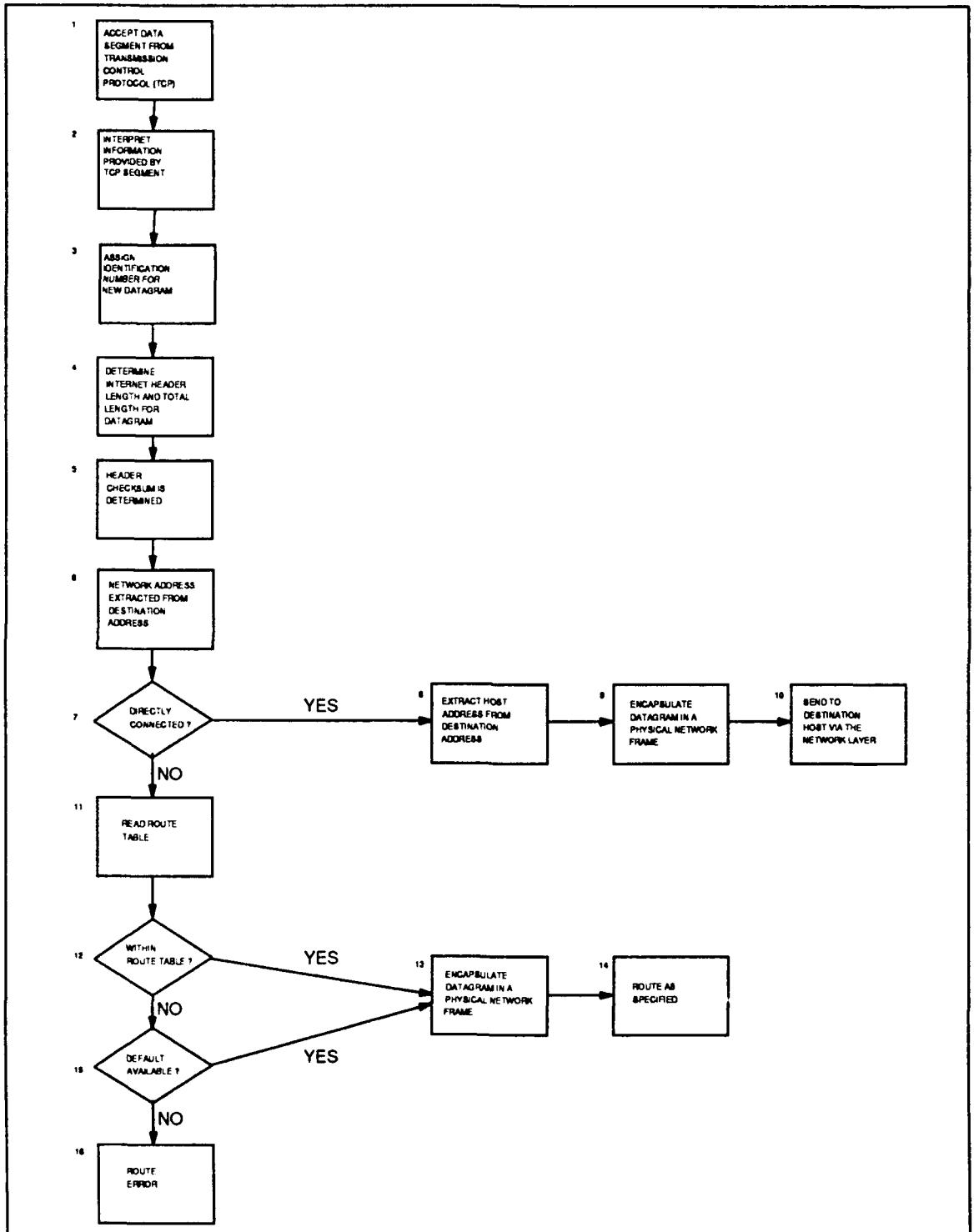


FIGURE 3.6 Internet Protocol Routing at a Source Host

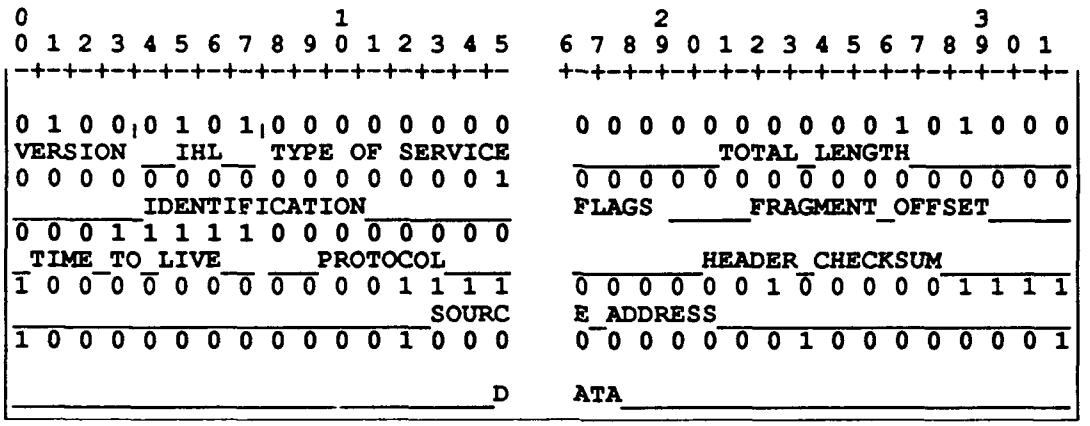
The header information of the datagram is divided into 16 bit words, summed using binary arithmetic, and the one's complement of the sum is used as the header checksum (5). Figure 3.7 illustrates an example of the computation of the header checksum of a datagram. Recall that the header checksum validates only the header information. The data portion is verified within the higher level TCP.

The network address can be easily extracted from the 32 bit destination address (6). All initial route decisions are based upon the destination network address.

If the destination network address is the same as the network address of the source host, then the hosts are said to be directly connected (7). Directly connected hosts use the Network layer to encapsulate the datagram and directly transfer the packet from the source host to the destination. First, the host address is extracted from the 32 bit destination address (8). The host address is the unique destination within the Network address. The datagram is encapsulated in a network packet (9). The network packet control information contains the host address. Subsequently, using the Network layer, the network packet is sent to the directly connected destination (10). The IP within the source host is complete.

Recalling the datagram format from Figure 3.2, the following example of a datagram with no options selected is provided to

illustrate the checksum. For simplicity, the datagram has been separated into 16 bit segments.



In this example, nine 16 bit words are summed using binary arithmetic and the one's complement is applied to this sum to obtain the header checksum of the datagram.

```
0 1 0 0 0 1 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1
0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 1
1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1
-----------------------------------
0 1 0 1 1 0 0 1 0 0 1 0 0 0 0 0 (Binary Sum)

1 0 1 0 0 1 1 0 1 1 0 1 1 1 1 1 (One's Complement of Binary Sum)
```

Therefore, the header checksum is the following 16 bit word:

1 0 1 0 0 1 1 0 1 1 0 1 1 1 1 1

Figure 3.7 Computation of a Header Checksum

If the source host and destination host are not directly connected, the destination is on a connected network and requires indirect routing. Each host has an internal route table within its IP module. The

destination network address is compares with the indicated values of the route table (11).

If the destination network address is within the route table (12), then a specific route for delivery is specified. The datagram is encapsulated in a network packet (13). The network packet control information contains the address of the subsequent gateway whose value was obtained within the route table. Using the Network layer, the network packet is sent to the appropriate gateway in route to its final destination (14). The IP within the source host is complete.

Usually, if the network address is not available within the host's route table, a default value is given to direct the datagram to a nearby gateway for further processing (15). If the network address is not within the source host's route table and a default value does not exist, then it is assumed that the destination is unreachable or invalid. A routing error is generated (16).

b. Gateway Routing

Within a gateway, the possibility exists that fragmentation of a datagram will be necessary. Figure 3.8 illustrates a flow chart of the IP processes that occur within a gateway. The numbers in parentheses correspond to the box numbers of Figure 3.8.

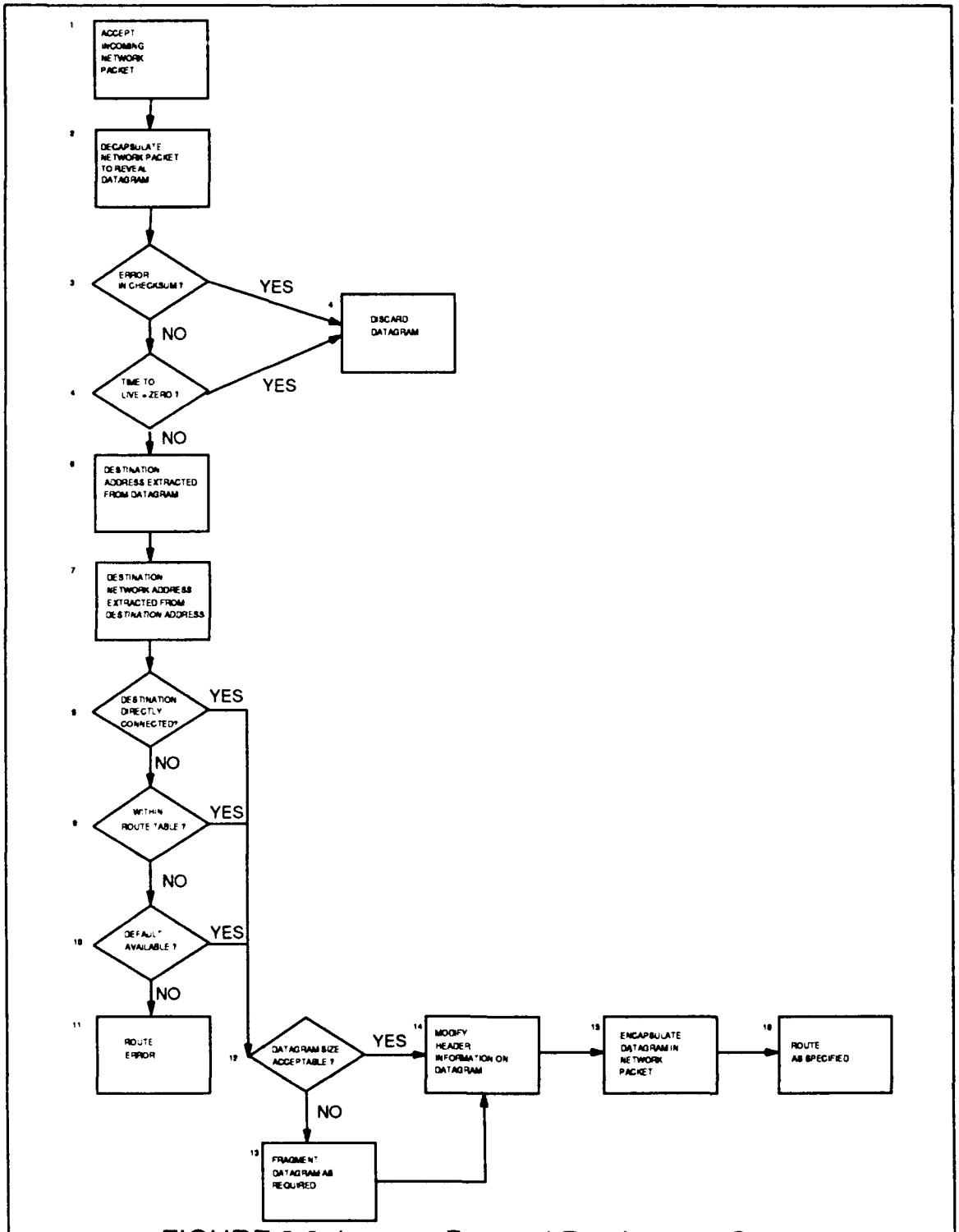


FIGURE 3.8 Internet Protocol Routing at a Gateway

The gateway will always receive the network packet of data from an adjacent network (1). The IP module within the gateway will decapsulate the network packet revealing a datagram (2). Immediately, the header checksum is computed at the gateway and compared with the value of the header checksum of the incoming datagram. If a problem exists with the header checksum (3) or if the TTL field has expired (4), the datagram is discarded (5). Recall that IP does not guarantee delivery of every datagram.

The 32 bit destination address is extracted from the datagram control information (6). The destination network address is extracted from within this destination address (7).

The gateway determines if the destination network is directly attached (8). If the destination is not directly attached, the gateway must evaluate its internal route table (9). If the destination network is not in the gateway's route table, a default value usually exists to direct the datagram to a subsequent gateway for further processing (10).

An error is indicated if the destination network is not attached to the gateway, not available within the route table, and no default exists within the route table (11). This error message indicates that the destination is unavailable or unreachable.

Only the IP module within a gateway may fragment a datagram. Fragmentation is necessary when a datagram originates in a local

network with a large packet size and must eventually pass through a network with a smaller packet size (12). Datagrams are fragmented as necessary in order to traverse the next network in route to its destination (13).

The header information of the datagram must be modified to reflect fragmentation (if any), a decrement in the TTL field, and a recalculation of the header checksum (14). The datagram is encapsulated within a network packet (15) to be delivered directly to its final destination or to a subsequent gateway (16).

c. Destination Host Routing

Within a destination host, the final stages of the IP communication process are initiated. Figure 3.9 illustrates a flow chart of the IP processes that occur within a destination host. The numbers in parentheses correspond to the box numbers of Figure 3.9.

The destination host receives the network packet of data (1) from an attached network. As in Figure 3.8, the IP module within the gateway will decapsulate the network packet revealing a datagram (2) and computes a header checksum to compare with the value of the header checksum of the incoming datagram (3). If a problem exists with the header checksum or if the TTL field has expired (4), the datagram is discarded (5) and an unsuccessful transmission has occurred.

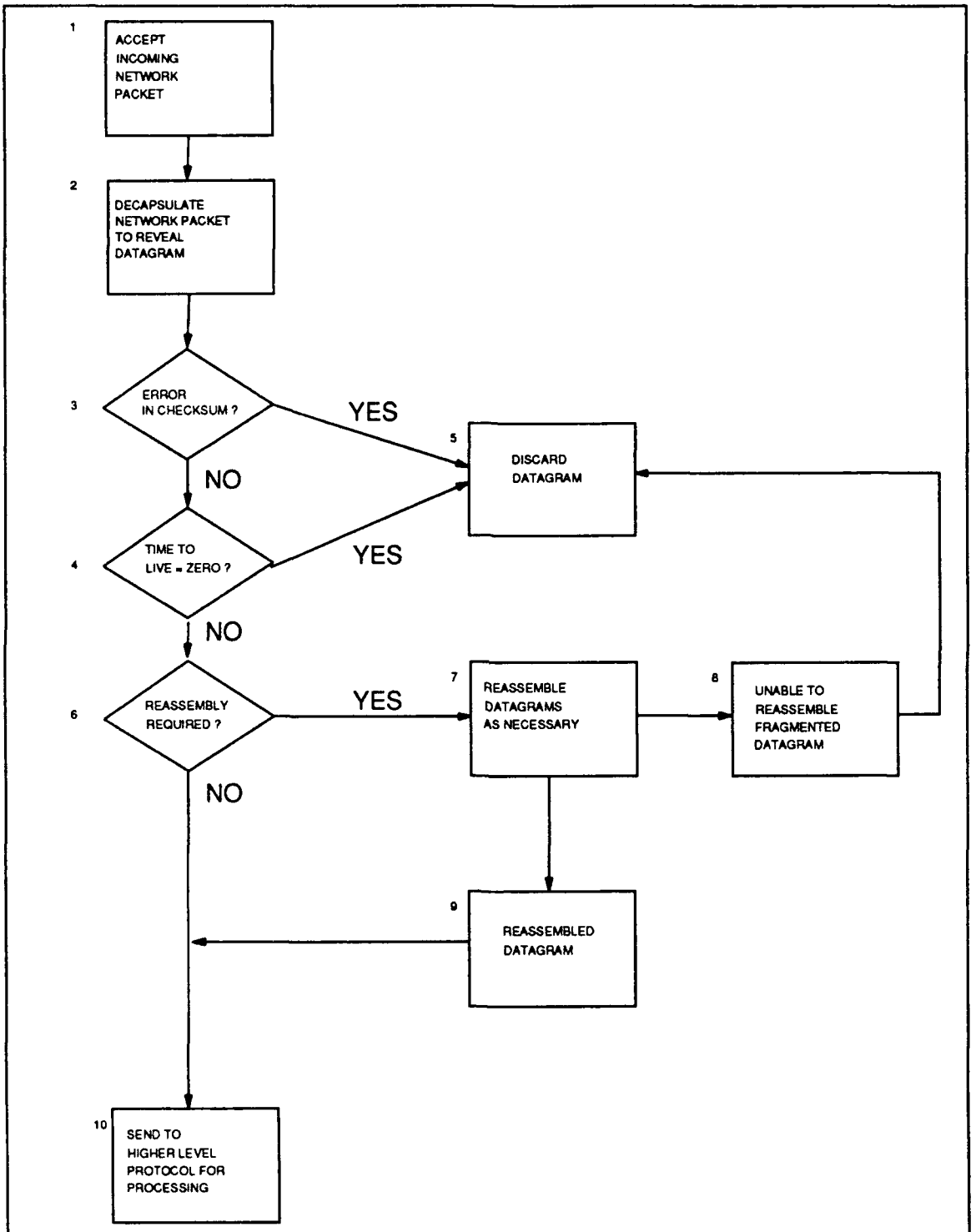


FIGURE 3.9 Internet Protocol Routing at Destination Host

Reassembly of fragmented datagrams may be necessary (6). The datagrams are reassembled as required (7). Because datagrams may be lost or garbled within the internetwork, it may be impossible to fully reassemble the original datagram within the time limits of the system. If a datagram cannot be reassembled within the limits of the system, all fragmented datagrams that were part of the original datagram are discarded (8).

Usually, all fragments are present and the reassembled datagram is available for further processing (9). Finally, the original datagram is sent to the next level protocol for processing (10).

D. FRAGMENTATION OF DATAGRAMS

1. The Need for Fragmentation

The individual networks that make up the internetwork are diverse and vary in capabilities. Specifically, networks may vary in the maximum acceptable packet size that may traverse over a network. Fragmentation becomes necessary when a datagram originates in a network with a large packet size and must traverse over a network with a smaller packet size. Fragmentation reduces the size of a datagram to an acceptable length so that the datagram may traverse the outgoing network.

Fragmentation of datagrams may only occur within IP modules of a gateway. Each fragmented datagram is treated as an independent datagram and thus may be further fragmented. All datagrams remain fragmented until they reach their final destination where reassembly occurs.

Each individual network of the internetwork has a maximum transmission unit (MTU) which determines the maximum size packet of information that can be transmitted within the network. A packet contains standard header information and a datagram. So the MTU indirectly determines the maximum size datagram that may be successfully transmitted over an outgoing network. The internetwork minimum value for the MTU is 576 bytes. However, many networks far exceed this value.

The decision to fragment is easy. If the incoming length of the datagram is greater than the size of the outgoing network's MTU, then fragmentation is necessary.

2. Header Information in Fragmented Datagrams

Fragmented datagrams have essentially the same header information as the original datagram. The differences between an original datagram and a fragmented datagram created from the original datagram are as follows:

1. The bits of the header that indicate that it is a fragment.
2. The bits indicating a change in the datagram's total length.
3. A recomputation of the header checksum.

Certain datagrams may be marked with a "don't fragment" field set within its header. This is set by the source host when it is known that the destination host does not have the required reassembly assets to combine the fragmented datagrams into the original datagram. Gateways will simply discard such datagrams if delivery cannot be achieved without fragmentation. A message is relayed to the source host by the Internet Control Message Processor (ICMP) indicating that the destination of the datagram is unreachable because fragmentation is necessary and the "don't fragment" flag has been set.

3. Fragmentation Procedure

Fragmentation decomposes a datagram into smaller pieces which are reassembled at the final destination. Recall that the source host assigns a unique integer to each datagram that is created. This identification number is unique for the time that the datagram will be active within the internetwork.

Subsequent fragmenting of a datagram maintains its original unique identification integer. This identification is used by the destination

host to distinguish fragments of one datagram from fragments of another datagram.

Figure 3.10 illustrates the current process of fragmentation that occurs in a gateway. The numbers in parentheses correspond to the box numbers of Figure 3.10.

A datagram arrives at an IP module of a gateway (1). This datagram may be identical to the datagram created at the source host or may be a fragment of the original datagram. The incoming datagram has previously established the value of its total length, its fragment offset, and its more flag identifier. These values are found within the header of the incoming datagram.

The more flag would have a value of zero if the datagram has never been fragmented or represented the last fragment of the original datagram. A value of one within the more flag would indicate that the original datagram had been fragmented and at least one datagram fragment will follow this datagram during reassembly. The fragment offset is used in conjunction with the total length to determine the exact position of the datagram during reassembly at the destination host.

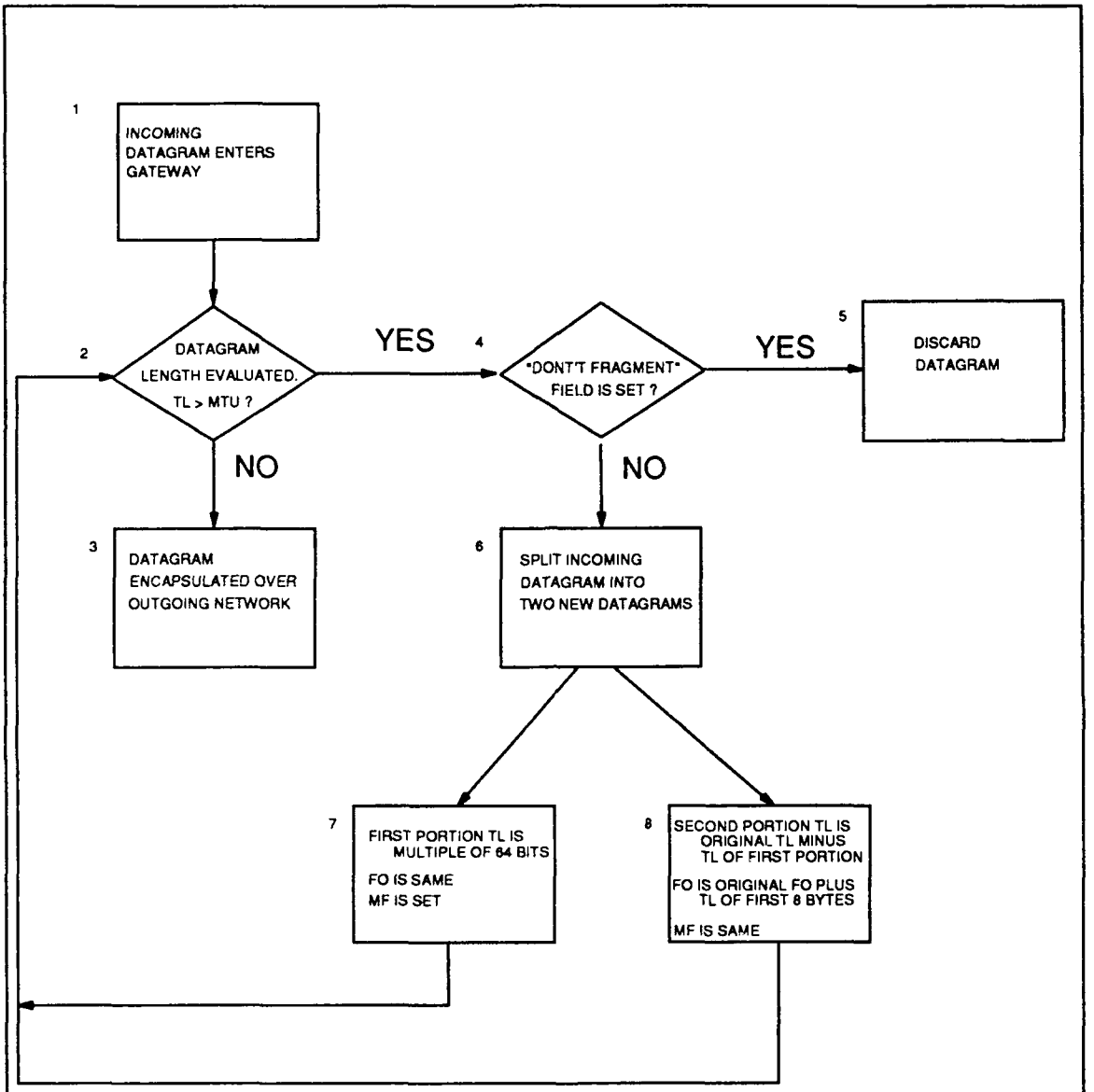


FIGURE 3.10 Internet Protocol Fragmentation Procedure

The total length (TL) of the datagram is evaluated (2). There is no need to fragment the datagram if the TL of the datagram is less than the outgoing network's maximum transmission unit (MTU). The datagram's header is adjusted as necessary and encapsulated within a network packet to be sent over the outgoing network. The header changes reflect the decrement of the TTL field, the necessary fragmentation, and the recomputation of the checksum (3).

If the TL of the datagram is greater than the MTU, then fragmentation is necessary for the successful transmission of the datagram across the outgoing network. Fragmentation of the datagram is not permitted under any circumstances if the "don't fragment" field is set within the datagram's header (4). Fragmentation is permitted if the "don't fragment field is not set.

The gateway will discard all datagrams where fragmentation is necessary for delivery but restricted by the "don't fragment" field within the datagram's header (5). A message is relayed to the source host by the ICMP indicating that the destination of the datagram is unreachable because fragmentation is necessary and the "don't fragment" flag has been set.

If fragmentation is permitted, two new datagrams are created to replace the incoming datagram (6). The header information of the incoming datagram is copied into the headers of the two new datagrams. This ensures

that the identification number of the incoming datagram is transferred to the two new datagrams as well as other pertinent information. Specific header information concerning fragmentation and the recomputation of the checksum will be adjusted as required.

The data of the incoming datagram is divided into approximately one half. The first portion is copied to the first new datagram. The remainder is copied to the second new datagram. A restriction applies to the length of the first portion of data. This data must have a length that is a multiple of 64 bits (8 bytes) (7). This is required because the fragment offset field of the datagram's header has a unit of measure of 64 bits. Therefore, the fragment offset of the second portion of data is directly related to the number of 64 bit length within the total length of the first datagram.

The first portion of the data has a length that is a multiple of 64 bits (8 bytes). The fragment offset will have the same value as the incoming datagram's first fragment offset. The more flag will be set on to indicate that a least one datagram fragment will follow this fragment during reassembly. The header information for this datagram is corrected to reflect these changes in values.

The second portion contains the remaining data (8). The TL of the header is adjusted to its new value. The fragment offset must be adjusted to reflect the offset caused by the fragmentation. The fragment offset of the

second portion is equal to the original datagram's fragment offset plus the quotient of the TL of the first portion in bytes divided by eight bytes (the unit of measure of the fragment offset). The more flag will have the same value as the incoming datagram. A zero value in the more flag field would indicate that this is the last datagram fragment of the original datagram. A one value in the more field indicates that at least one datagram fragment will follow this datagram in reassembly.

The two new datagrams are the recycled through the process of the fragmentation procedure for further evaluation. A recursive fragmentation process occurs until the fragments are of an acceptable length to the outgoing network.

4. Proposed Improvement to the Fragmentation Process

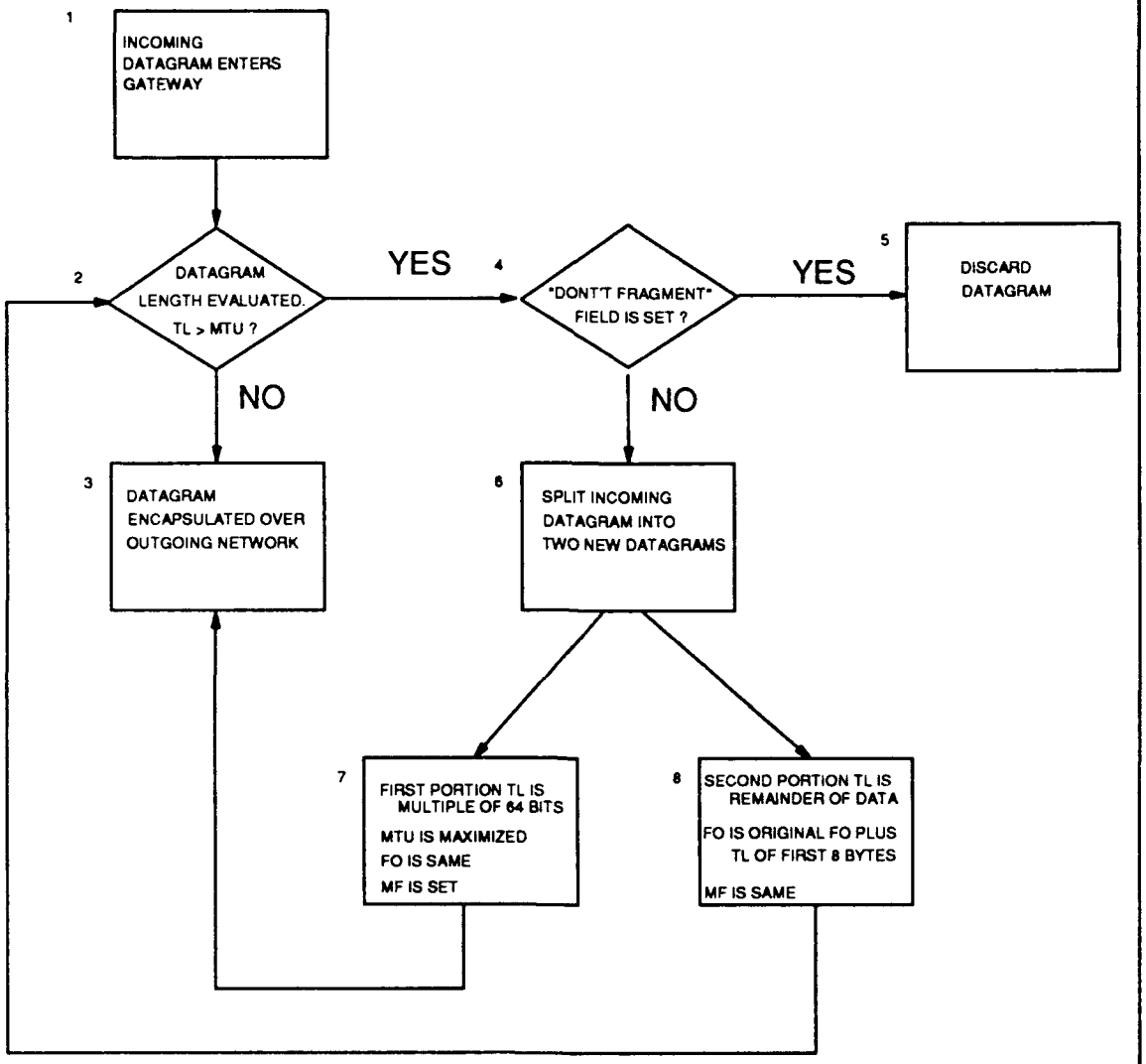
The present algorithm for fragmentation could be improved upon to maximize throughput and to decrease congestion within the internetwork. Improvements could be realized by maximizing the amount of data that will be contained in the packet of an outgoing network.

The same processing resources are associated with handling a packet that is empty, half full, or completely full. Under specific conditions, the amount of packets sent over an outgoing network could be reduced by modifying the fragmentation algorithm so that full or nearly full packets of information would flow over the outgoing network as often as possible.

The current fragmentation algorithm divides the incoming datagram's data into two approximately equal parts and then evaluates each new datagram against the outgoing network's MTU. The recommended improvement to this algorithm would be to ensure that the first datagram fragment maximizes the potential of the outgoing network's MTU. The MTU includes the bytes required in the packet header, the datagram header, and the data within the datagram. The idea of the improved algorithm is to calculate the amount of space remaining in the packet after allowances have been made for packet header and the datagram header. Once the remaining space has been calculated, the data from the incoming datagram is copied into the first datagram fragment in factors of 64 bits. The limit of space remaining is approached by the transferred data but it is never exceeded. Once the datagram is full, it is forwarded over the outgoing network.

The second datagram fragment contains the remaining data. Only this datagram recurses through the system for possible additional fragmentation.

Figure 3.11 illustrates a flow chart of the proposed improvements to the fragmentation algorithm. Notice the similarities to Figure 3.10 with the exception of blocks six, seven and eight. The numbers in parentheses correspond to box numbers of Figure 3.11.



MTU - MAXIMUN TRANSMISSION UNIT

TL - TOTAL LENGTH OF DATA WITHIN THE DATAGRAM

FO - FRAGMENT OFFSET WITHIN THE DATAGRAM

MF - MORE FLAG INDICATOR WITHIN THE DATAGRAM

FIGURE 3.11 Proposed Improvement to the Present Fragmentation Algorithm

Initially, The datagram is handled identically to the current process described in Figure 3.10. The improvement occurs during the split of the incoming datagram (6). Two new datagrams are also created and the header information of the incoming datagram is copied into the headers of the two new datagrams.

Utilization of resources is improved when the amount of data loaded into the first new datagram is maximized.Suppose the first new datagram evaluates the MTU of the outgoing network, makes allowance for the length of the packet header and the datagram header, and computes the remaining space available within the packet for actual data. Data is copied from the incoming datagram in whole factors of 64 bits. The remaining space available for data may be maximized but not exceeded. Recall that the 64 bit factor is required because the fragment offset of the second portion of data is directly related to the number of 64 bit lengths within the TL of the first datagram.

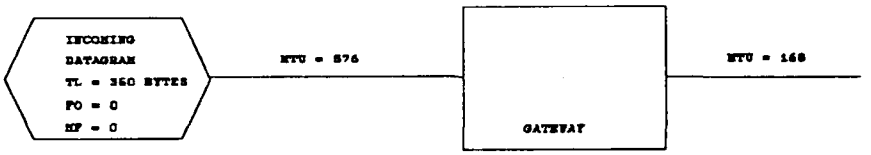
This datagram will therefore meet the specifications of the outgoing network's MTU. Further fragmentation of this datagram will not be required at this gateway(7). The datagram's header is adjusted as necessary and encapsulated within a network packet to be sent over the outgoing network. The header changes reflect the decrement of the TTL field, the necessary fragmentation, and the recomputation of the checksum.

The remaining data is copied to the second new datagram (8). This second datagram must eventually be evaluated to determine if further fragmentation is necessary. The TL of the header is adjusted to its new value. The fragment offset must be adjusted to reflect the offset caused by the fragmentation. The more flag will have the same value as the incoming datagram.

Only the second datagram is recycled through the fragmentation process for further evaluation. A recursive fragmentation process occurs until the fragments are of an acceptable length to the outgoing network.

5. Comparison of the Existing Fragmentation Procedure and the Proposed Fragmentation Procedure

A comparison of an example of fragmenting datagrams with the current method and the proposed improved method is given in Figures 3.12 and 3.13. In this particular example, there is a reduction of one packet of data using the improved algorithm.



Given the following information:

Incoming network MTU = 576 bytes.
Outgoing network MTU = 168 bytes.
Incoming datagram TL = 350 bytes.
Incoming datagram FO = 0.
Incoming datagram MF = 0.
Fragmentation of the datagram is permissible.

Using the flow chart of Figure 3.10, the following steps of the flow chart are encountered in the current fragmentation process:

1. The incoming datagram enters the gateway.

2. Fragmentation is necessary. (TL > MTU)

3. Not applicable.

4. Fragmentation is permissible.

5. Not applicable.

6. The incoming datagram is replaced by two new datagrams. The header information of the incoming datagram is copied into the header information of each datagram. The data of the incoming datagram is divided into approximately equal parts. A restriction applies to the length of the first portion which requires that the TL of the first portion be a multiple of 64 bits (8 bytes).

7. The first datagram will have the following values:

   TL = 176 bytes (multiple of 64 bits).
   FO = same as incoming datagram (0).
   MF = 1 (indicating additional fragment(s) will follow)

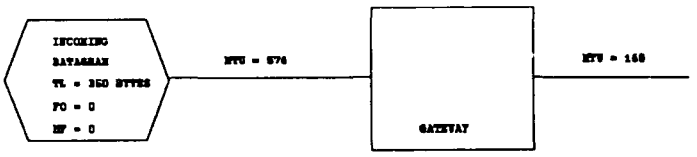
8. The second datagram will have the following values:

   TL = 350 bytes (original TL) - 176 bytes (TL of first datagram) = 174 bytes.
   FO = 0 (original FO) + (176 bytes (TL of first datagram)/8 bytes (unit of measure of FO)) = 22.
   MF = same as incoming datagram (0).

Each of these new datagrams is still too large for the outgoing network's MTU. Therefore, each datagram must recurse through the fragmentation procedure. This yields the following four acceptable datagrams that must be encapsulated in a network packet:

Datagram 1A	Datagram 1B	Datagram 2A	Datagram 2B
TL = 88 bytes	TL = 88 bytes	TL = 88 bytes	TL = 86 bytes
FO = 0	FO = 11	FO = 22	FO = 33
MF = 1	MF = 1	MF = 1	MF = 0

Figure 3.12 Example of Fragmenting a Datagram Using the Present Algorithm



Given the following information:

Incoming network MTU = 576 bytes.
Outgoing network MTU = 168 bytes.
Incoming datagram TL = 350 bytes.
Incoming datagram FO = 0.
Incoming datagram MF = 0.
Fragmentation of the datagram is permissible.

Using the flow chart of Figure 3.11, the following steps of the flow chart are encountered in the modified fragmentation process:

1. The incoming datagram enters the gateway.
2. Fragmentation is necessary. (TL > MTU)
3. Not applicable.
4. Fragmentation is permissible.
5. Not applicable.

6. The incoming datagram is replaced by two new datagrams.The header information of the incoming datagram is copied into the header information of each datagram. The data of the incoming datagram is divided into two parts. The first fragmented datagram will take full advantage of the space available in the outgoing network's MTU by copying data in 64 bit intervals until the space provided is approached but not exceeded.

7. The first datagram will have the following values:The MTU is 168 bytes and we will assume 40 bytes are required for the packet header and the datagram header. The original datagram had 350 bytes. The value of the MTU is known (168 bytes) and 40 bytes are required for the header information leaving 128 bytes of available space. The entire length of the 128 bytes can be used for data because 128 bytes is a multiple of 64 bits. This datagram may now be successfully processed by the outgoing network.

Datagram 1
TL = 128 bytes (multiple of 64 bits).
FO = same as incoming datagram (0).
MF = 1 (indicating additional fragment(s) will follow)

8. The second datagram will have the following values:

TL = 350 bytes (original TL) - 128 bytes (TL of first datagram) = 222 bytes.
FO = 0 (original FO) + (128 bytes (TL of first datagram)/8 bytes (unit of measure of FO)) = 16.
MF = same as incoming datagram (0).

This remaining datagram is too large for the outgoing network's MTU and must be fragmented using the same algorithm. This replaces the second datagram with two new datagrams with the following header information:

Datagram 2	Datagram 3
TL = 128 bytes (multiple of 64 bits	TL = 222 bytes - 128 bytes = 94 Bytes
FO = same as incoming datagram (16)	FO = 32
MF = 1	MF = 0

Hence, only three packets are generated.

Figure 3.13 Example of Fragmenting a Datagram Using the Proposed Modification to the Fragmentation Algorithm

Clearly, a datagram may be fragmented several times before its TL is capable of traversing an outgoing network's MTU. Extraneous packets reduce system performance and add to the congestion within the internetwork. Congestion is a condition of severe delay caused by an overload of datagrams within the internetwork. As the use and availability of the military internetwork expands, the problems associated with congestion increases as well.

The proposed improvement to the fragmentation algorithm has the potential to reduce the number of fragmented datagrams created within the internetwork. The proposed algorithm simply maximizes the outgoing network's MTU assuring maximum throughput and efficiency. Thus, by modifying the fragmentation algorithm and maintaining all other aspects of the IP, the system performance may be enhanced.

E. REASSEMBLY OF DATAGRAMS

Reassembly of fragmented datagrams occurs only within the IP module of the destination host. The identification field, assigned by the source host, is used to distinguish fragments of one datagram from fragments of another datagram.

The IP module of the destination host checks every incoming datagram to determine if reassembly is necessary. An examination of the MF and FO of

the incoming datagram's header can immediately make this determination. If an unfragmented datagram arrives at the destination, it is immediately passed to the next protocol for further processing.

The IP module within the destination host must buffer the incoming datagram fragments until the original datagram can be reassembled or until its reassembly timer has expired. Once reassembly of the original datagram is complete, the data is passed to the next highest protocol for further processing.

Reassembly of fragmented datagrams is complicated by the fact that each fragment is regarded as an independent datagram. Hence, delivery is not guaranteed. Each fragment may take alternative routes to the destination, may be lost, arrive out of order, or may be internally garbled. If the reassembly timer expires before reassembly has been completed, all fragments of data that are partially reassembled are discarded without processing.

Sometimes, because of various system delays, duplicate data may be created and received. All duplicate datagrams that are received by the destination host are discarded. Duplicate datagram fragments may be the fragment that was sent first but, because of system delays, arrived after the receipt of the retransmission of the fragmented datagram.

A buffer identification is created for each arriving datagram. This identification is a concatenation of the source address, destination address,

protocol, and identification fields of the incoming datagram's header. The buffer identification ensures that fragmented datagrams are segregated to their proper reassembly resources. Reassembly resources will be allocated within the destination host for all unique buffer identifications. Reassembly resources include a data buffer, header buffer, total data length field, reassembly timer, and a reassembly fragment bit block table. A destination host has the capability of simultaneously reassembling more than one original datagram.

The reassembly resource data buffer is a finite size buffer that is created to temporarily store data from associated fragments. Data is placed in the buffer relative to the FO and the TL of the associated fragmented datagram.

The reassembly resource header buffer is a temporary buffer that is established to hold the header information of a datagram that is undergoing reassembly. Header information is copied from the fragment that represents the first portion of the fragmented datagram (FO = 0) to the header buffer. Specific information of the header concerning fragmentation and TL will be adjusted as necessary prior to further processing.

The TL of the reassembled datagram must be calculated to enable further processing. The reassembly resource total data length can be calculated once the fragment that represents the last portion of the fragmented datagram (MF = 0) has been received. Recall that the FO is in units of 64 bits

or 8 bytes and that the Internet Header Length (IHL) is in units of 32 bits or 4 bytes. The total data length is computed using the formula described in Figure 3.14.

$$TDL = (TL + (FO * 8) - IHL * 4)$$

TDL = Total Data Length
TL = Total Length of Datagram
FO = Fragment Offset of Datagram
IHL = Internet Header Length of Datagram

Figure 3.14 Computation of Total Data Length of Reassembled Datagram

The reassembly resource timer exists in addition to the TTL associated with each datagram. This timer is unique to the destination host and is directly related to the buffer capacity available and the data rate of the transmission medium. So the maximum value for the timer is determined by the quotient of the buffer size divided by the transmission rate [Ref. 8]. Figure 3.15 provides an example of the maximum reassembly timer:

Given the following information:

Buffer Size: 200 Kb
Transmission Rate: 10 Kb/sec

Determine the Maximum Reassembly Timer.

$$\text{Maximum Timer} = \frac{\text{Buffer Size}}{\text{Transmission Rate}} = \frac{200\ \text{Kb}}{10\ \text{Kb/sec}} = 20\ \text{sec}$$

Figure 3.15 Example of Maximum Reassembly Timer Computation

The reassembly resource fragment bit block table is a bookkeeping device that indicates missing fragments in reassembly and can indicate when reassembly is complete. Zero represents the beginning of the fragment bit block table and the fragment that represents the last portion of the original datagram (MF = 0) determines the end of the bit block table. Once a fragment has been accepted, the fragment bit block table is acknowledged and indicates that data covering a specific range has been received. A whole reassembled datagram exists when the fragment bit block table is full.

Figure 3.16 illustrates a flow chart of the reassembly process. The numbers in parentheses correspond to the block numbers of Figure 3.16.

Reassembly may be determined to be necessary at the destination host. A datagram arrives at the destination host (1). This datagram may be a fragmented datagram or a whole datagram. Upon the arrival of each datagram, a buffer identification number is computed (2). This number is a concatenation of the source address, destination address, protocol, and identification fields found within the datagram's header.

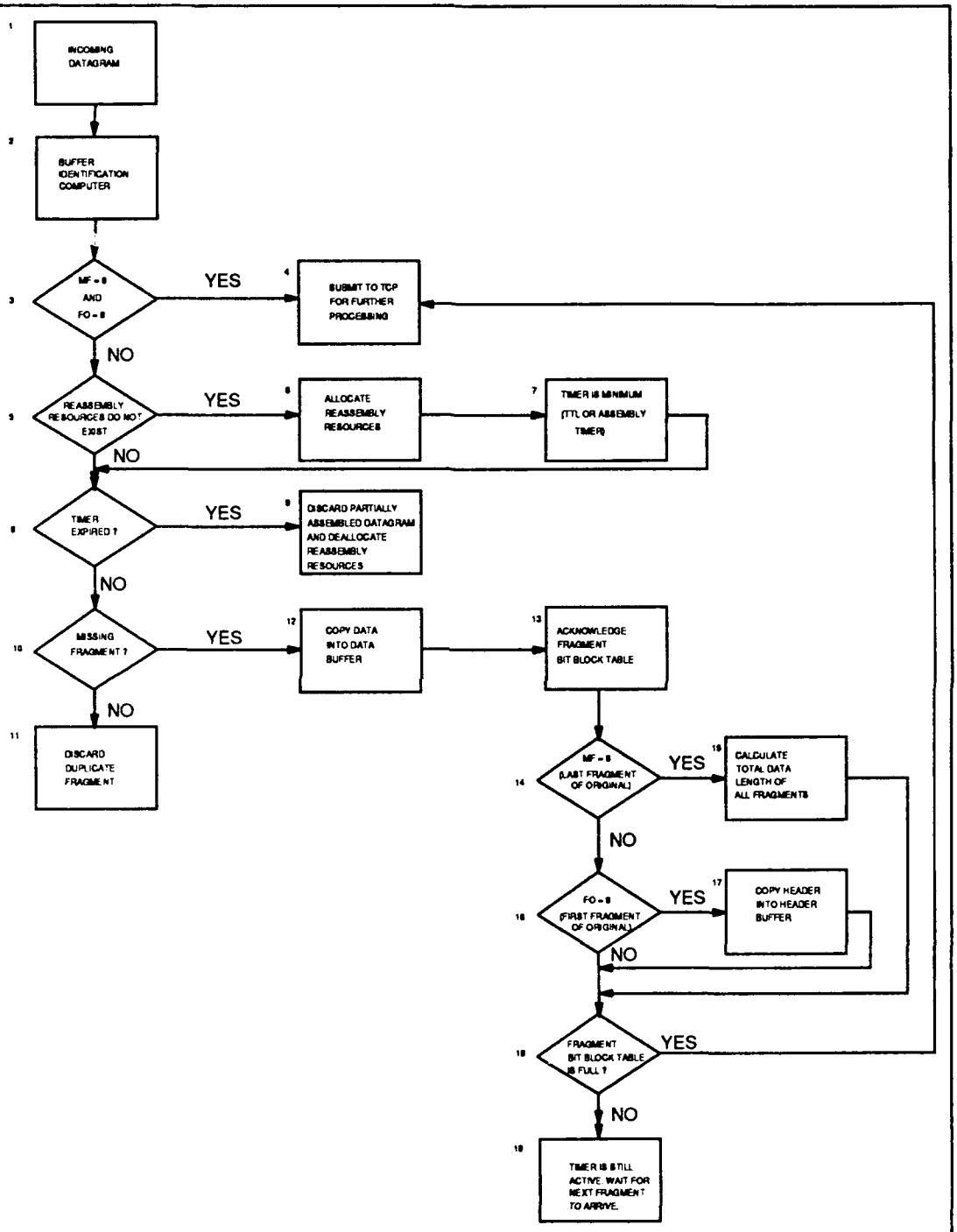


FIGURE 3.16 Reassembly of Fragmented Datagrams

A test is performed to determine if the incoming datagram is a fragmented datagram or whole datagram. A whole datagram has never been fragmented and can be determined if the incoming datagram's MF field and FO field both have the value of zero (3).If a whole datagram exists (4), this datagram can be submitted to a higher level protocol (usually TCP) for further processing.

Reassembly resources are allocated for each individual buffer identification number. Reassembly resources must be established if the incoming datagram's buffer identification number does not represent a previously established resource (5).

Reassembly resources include a data buffer, header buffer, total data length field, reassembly timer, and a fragment bit block table (6). Initially, the reassembly timer is set to the minimum of the values represented by the TTL field of the datagram and value of the maximum reassembly timer value of the associated destination host (7). Figure 3.15 discusses the maximum reassembly timer value of the associated destination.

The reassembly timer is examined to ensure that time is remaining in the reassembly process (8). The partially assembled datagram with the specific buffer identification is discarded and all reassembly resources are released if the reassembly timer expires (9).

An incoming fragmented datagram may be required for successful reassembly or the incoming fragmented datagram may be a duplicate datagram (10). All duplicate datagrams are discarded by the IP module (11).

If the datagram is required for reassembly, the data portion of the fragmented datagram is copied into the data buffer relative to the datagram's FO and TL (12). The fragment bit block table is acknowledged to indicate the receipt of data (13).

If the datagram's MF field has a value of zero, no other fragment will have a relative position after this fragment. This is the last portion of the original datagram (14). The total data length of the complete datagram can then be calculated (15).

If the FO has a value of zero, no other fragment will have a relative position before this fragment. This is the first portion of the original datagram (16). The header information of the first portion of the original datagram is then copied into the header buffer (17).

A determination is made if reassembly is complete. A full fragment bit block table indicates a fully reassembled datagram (18). Sometimes, a partially reassembled datagram exists within the reassembly resources. The timer continues to decrement as the destination awaits further fragments (19).

F. SUMMARY

IP is one of the vital protocols that allow individual networks to function as a single unified network within the military. Underlying details of IP are hidden from the user.

IP coordinates with TCP and the Network layer to successfully transfer data. IP accepts data from an application program via the TCP and uses the Network layer during the actual communication between connected computers.

Gateways are computers that provide a physical connection between networks. The IP module within a gateway is responsible for making a routing decision based upon the datagram's address and what it knows about the connectivity of connected networks. Because of the varying capabilities of networks, fragmentation may be necessary in order to reduce a datagram to a length that can be successfully handled by an attached network. Reassembly of the fragmented datagram occurs at the final destination host computer.

IP uses datagrams to transfer data. Each datagram is treated individually by the military internetwork. IP is generally reliable, but successful delivery of each datagram is not guaranteed. Should a datagram be lost or garbled, the higher level TCP will eventually discover the problem and retransmit the data.

Overall, IP is an efficient and successful protocol. However, it was discovered that an improvement in efficiency can be realized by slightly

modifying the IP fragmentation procedures. This improvement is illustrated in sections D.4 and D.5 of this chapter.

IV. TRANSMISSION CONTROL PROTOCOL (TCP)

A. INTRODUCTION

TCP is an example of a transport layer that is standard for all Department of Defense networks. Its purpose is to provide a reliable means for two connected hosts to exchange data. TCP ensures data is delivered in the proper sequence without errors, loss, or duplication of data. This reliability is achieved despite the fact that the two hosts may be located across various unreliable networks.

TCP cannot exchange data directly between the source host and the destination host. It must utilize the unreliable lower level Internet Protocol (IP) and Network layer to provide reliable data transport. Fig 2.3 illustrates data encapsulation and the dependency TCP has on the lower protocols.

Coordination must be maintained between the TCP and the application program. Arbitrarily long messages are passed from the application program and buffered within the TCP's send buffer. The TCP packages the data within the send buffer to form a segment. The appropriate header information is added to this segment and this segment becomes the data portion of the IP datagram. An original IP datagram is associated with each TCP segment.

Segments are the basic unit of transfer within the TCP. They can be exchanged in order to establish a connection, transfer data, transfer acknowledgments, or to close a connection.

TCP establishes and synchronizes the connection between the source and the destination host. It is responsible for maintaining this connection as well as terminating this connection at an appropriate time.

Reliability is maintained because the TCP can be depended upon for error control of the data, flow control of the data, and for security and precedence. Error control is accomplished by maintaining a checksum on a combination of the segment's header and data portion. Recall that IP's checksum only verified the datagram's header and disregarded verification of the data portion of the datagram.

Flow control is maintained by utilizing a variable sliding window method to control the data rate between two connected hosts. TCP establishes the security and precedence at the initial connection establishment of the source host and destination host. This information is relayed to the IP and it is enforced to the best of IP's capabilities.

As with IP, the underlying details of the TCP are hidden from the user. TCP cannot be accomplished without IP and the integrity of the information transported and delivered by IP cannot be assured without TCP. Consistently,

IP and TCP are often referred together as TCP/IP rather than as individual entities. The following sections examine the TCP in detail.

B. FORMAT OF TCP SEGMENTS

Each segment of data within TCP consists of data from the application program concatenated with control information. This control information is universally understood within the TCP layer and is necessary for reliable communication. This section details the specific entities that combine to form a complete segment.

Once a connection has been established, a full duplex connection exists between the source host and the destination host. The segment's control information is arranged in such a manner that it can be used universally by both the source host, when sending data, and the destination host, when sending acknowledgments.

This standardization of the segment's control information provides for system design simplicity but adds to the network congestion by sending irrelevant bytes of information with each segment. As with the IP datagram, the TCP segment will always contain header information that will be a multiple of 32 bits. Figure 4.1 illustrates the segment format [Ref 9].

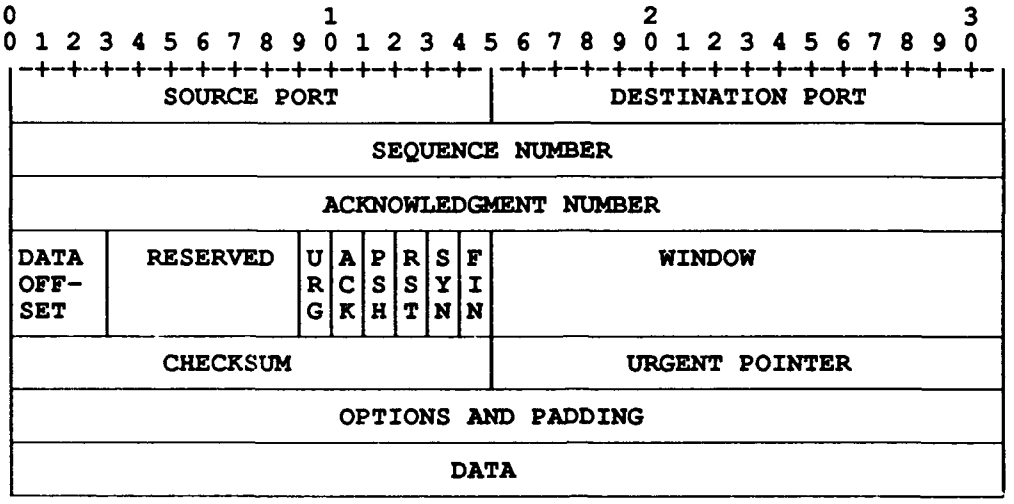


Figure 4.1 Format for TCP Segments

The first 16 bytes of the segment's header indicate the unique port of the source host. The port can be considered as an access point to the host. The next 16 bytes indicate the unique port of the destination host.

The second 32 bit word indicates the segment's SEQUENCE NUMBER. This sequence number is used to ensure reliability of the data transfer. The TCP must ensure that data is not damaged, lost, duplicated, or delivered out of sequence. Each byte of data sent from the source has an associated sequence number that requires a positive acknowledgment from the destination's host. Sequence numbers are used at the destination to correctly arrange segments that may have been received out of order and is also used to discard duplicate data. Details of sequencing are discussed in Section D of this chapter.

The ACKNOWLEDGMENT NUMBER is the next 32 bit word. This acknowledgment number is sent by the destination host and indicates the next sequence number that it expects to receive.

The DATA OFFSET field is a four bit long field that indicates the number of 32 bit words within the control information of the segment. The control information is always a multiple of 32 bit words. The data offset is used to calculate the position within the segment where data begins.

The following six bits are reserved for future use. A zero currently occupies each bit of this field.

The URG, ACK, PSH, RST, SYN, and FIN fields all contain a single bit. The URG field signifies whether or not the accompanying data is urgent. Some data may have special urgency. TCP will attempt to transfer the data as rapidly as possible if this field is set. An interrupt or break would set the URG field. This field also indicates to the destination host that urgent data is in the incoming data stream.

The ACK field signifies that a segment is an acknowledgment and the acknowledgment number is significant. The PSH field indicates a 'push' if it is set. Usually, data accumulates from the application program within the TCP send buffer. The TCP usually packages this data, at its own discretion, into segments. If a PSH is set, the protocol will send and receive the data

without regards to whether the buffer is full. This is usually used during connection termination.

The RST field indicates that the connection is being reset. A RST must be sent whenever a segment arrives which apparently is not intended for the current connection. The SYN field is used in the initial connection establishment to synchronize the sequence numbers expected between the source host and destination host. Since connections must be established between unreliable networks, this synchronization is necessary to avoid erroneous data transfer. Lastly, the FIN field used to indicate that no more data will be sent by the sender. It will indicate the last segment that will be sent by the sender unless a retransmission of an earlier segment is required.

The WINDOW is a 16 bit field that effects flow control. The window is used by the receiver to regulate the amount of data that is sent by the sender. With every acknowledgment, the window field is set to indicate the number of bytes that the sender may transmit before receiving further permission. Details of sequencing are discussed in Section E of this chapter.

The CHECKSUM ensures the validity of the header information and the data portion of the segment. The TCP checksum is computed very similarly to the way that the IP checksum is computed. This is illustrated in Figure 3.7. The segment is broken into 16 bit words, a binary sum of the 16 bit words is computed and a one's complement of the sum determines the value for this 16

bit field. Zeros are used, if necessary, to pad the bytes required to determine a 16 bit word.

The URGENT POINTER is used only if the URG field is set. It indicates the offset from the sequence number, within this segment, where the urgent data exists.

The OPTIONS field is used for miscellaneous purposes. It may communicate buffer sizes during setup procedures or may determine the maximum segment size that can be accepted by the destination. The OPTIONS field may contain zero or more 32 bit words. The PADDING ensures the OPTIONS field is a 32 bit word by filling the remaining spaces with zeros. The remaining segment contains the DATA from the application program.

C. CONNECTION ESTABLISHMENT AND CONNECTION TERMINATION

In order for two processes to communicate, their respective TCP programs must first establish a connection. Once the connection is established, data can flow between the two processes simultaneously.

A connection is established between a pair of sockets. Recall that a socket is a concatenation of the network identification, host identification, and

port identification of the host. A given socket may have more than one connection but only one connection to a specific remote socket.

During connection establishment, characteristics of data transfer, such as security and precedence, are agreed upon. Connection establishment also enables the TCP of the source host and destination host to maintain information concerning the actual connection such as relevant sequence numbers and relevant acknowledgment numbers. Reliability and flow control require the initialization and maintenance of this information.

Connections must be established over unreliable networks using IP. Messages may be lost or garbled during transmission across the interconnected networks. To ensure connection establishment over this unreliable network, TCP uses a mechanism called a three way handshake.

The three way handshake enables the TCP within the hosts to be synchronized. Synchronization is accomplished by exchanging segments that contain the SYN flag and an initial sequence number (ISN). ISNs are determined by an internal 32 bit clock within the sending host. The source host and the destination host will probably have different ISNs for their transmissions because the ISN is based on an internal clock unique to that host. Figure 4.2 illustrates the concept of the three way handshake. [Ref 10].

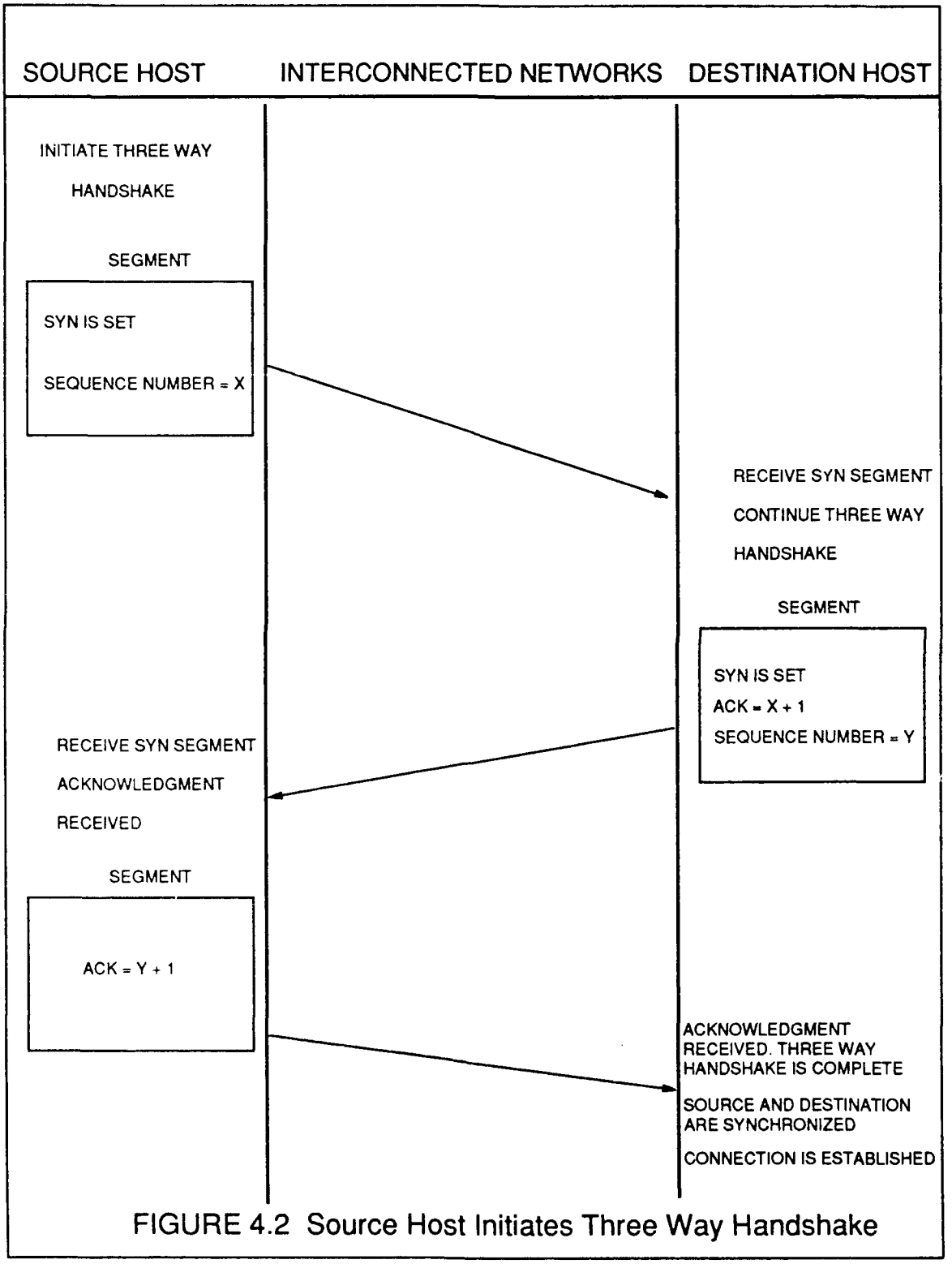


FIGURE 4.2 Source Host Initiates Three Way Handshake

Figure 4.2 identifies the source host as initiating the three way handshake. A segment is sent to the destination with the SYN flag set and an internally distinguished ISN (X). The destination will send a segment back to the source with its SYN flag set. This segment will be an acknowledgment of the sources ISN, (X + 1), and this segment will have an ISN of its own that is determined by its internal 32 bit clock (Y). The final message is only an acknowledgment of the destination's ISN (Y+1). Its purpose is to inform the destination that both sides agree that a connection has been established.

Once synchronization is complete, the source and destination TCP have coordinated the expectant sequence numbers and both are ready to transfer and receive data. It is possible that data may be exchanged during synchronization. In such cases, the TCP must hold the data until the handshake is complete. After the connection has been established, the application program will have access to the received data.

Once a connection is established, data may flow in both directions. To terminate such a connection, each TCP must issue a close primitive. Usually, this occurs when the TCP of the source host has no more data to send. A final segment with the FIN flag set is sent to the destination host to indicate that no new data will be sent. The advantage of this type of closure is that the connection will remain active until the destination host returns a segment with the FIN flag set indicating that no more acknowledgments will be sent. Using

this method, no data will be lost in transit because data will be retransmitted as necessary until all segments have been acknowledged.

System failure or the user cancellation of a program may cause an abrupt termination. During an abrupt termination, no more data is accepted and the connection is deleted. Using this method, data may be lost in transit.

Every byte of data transmitted by TCP has its own private sequence number. Connection establishment ensures the synchronization of these numbers between the source host and the destination host. The sequence numbers are crucial to the flow control and the reliability of TCP. Connection termination is important in freeing the sockets so that future communications are possible.

D. RELIABILITY

TCP has the ability to recover from data that is lost, duplicated, erroneous, or delivered out of order. This reliability is achieved because of TCP's use of sequence numbers that are assigned to each byte of data transmitted by the sender.

It is fundamental to TCP that every byte of data sent within a segment contains its own sequence number. The sequence number is a 32 bit field found within the control information of the transmitted segment. Because it

is a 32 bit number, there are 4,294,967,296 different values that can be associated with this field.

Within a segment, the first byte of data is identified directly by the sequence number indicated in the control information of the segment. The sequence numbers of the remaining bytes of data are implied by the length of data in the segment. Figure 4.3 illustrates the determination of sequence numbers:

Given: Data Segment
Length of data: 9 bytes
Actual Data: "TEST DATA"
Segment Sequence Number: 100

DATA:	T	E	S	T		D	A	T	A
SEQUENCE NUMBER:	100	101	102	103	104	105	106	107	108

The first byte 'T' is directly indicated by the sequence numbers. The remaining bytes of data have sequence numbers that are implied by the length of data.

Figure 4.3 Example of Sequence Number Determination

An initial 32 bit sequence number is determined by the sender during the initial connection between the source and the destination. This initial 32 bit sequence number is bound to a 32 bit clock that increments approximately every four microseconds. It is important to note that the 32 bit clock is unique to the sender and is not universal to the entire internet [Ref 10]. This internal clock cycles completely ever four hours and 33 minutes. Because of this delay

between repeating sequence numbers, each sequence number in transit is assumed to be unique for a particular connection.

When a TCP segment containing data is transmitted, a copy of this segment is placed in a retransmission buffer and a timer is initiated. The destination TCP must send a positive acknowledgment of receipt of the incoming sequence numbers within the designated timeout period or the sender will retransmit the segment. This acknowledgment is very similar to acknowledgments sent in sliding windows protocol because the acknowledgment may be cumulative instead of an individual acknowledgment for each byte sent.

The acknowledgment is essential to TCP. The acknowledgment is the sequence number of the first byte of data that it has not received. This means that all sequence numbers up to, but not including, the acknowledgment are considered as received. So the acknowledgment is actually the next sequence number expected by the destination. A window of the maximum bytes the sender can send without further permission is included within each acknowledgment. The window is a flow control mechanism and will be discussed in Section E of this thesis.

Not only are sequence numbers used for acknowledgment purposes, but the sequence numbers are used by the destination to correctly order segments and to ensure that duplicates are not processed. If segments of data are ever

lost or discarded, the sender will simply retransmit the segment until an acknowledgment is received or until the connection is terminated. Reliability is also enhanced by the use of a checksum that validates both the control information and the data of a segment.

Figure 4.4 illustrates the source TCP actions regarding sequence numbers and acknowledgments. The following numbers in parentheses correspond to the box numbers of Figure 4.4.

Initially, a connection is established between the two TCP entities of the host and the destination. The initial sequence number is synchronized and the length of an acceptable segment is agreed upon. The host TCP coordinates with the application program to handle the data transfer of an arbitrarily long message to the destination TCP (1). The TCP encapsulates the data and adds the proper control information to create a segment (2).

The segment is then passed to the IP where it will become the data portion of a datagram (3). A copy of the segment will be placed into a retransmission queue buffer and a timer will be initiated (4).

If the retransmission timer expires (5), the expired segment within the retransmission queue buffer will be sent again. If an acknowledgment is received prior to the timer's expiration, the acknowledged segment will be deleted from the retransmission queue buffer (6).

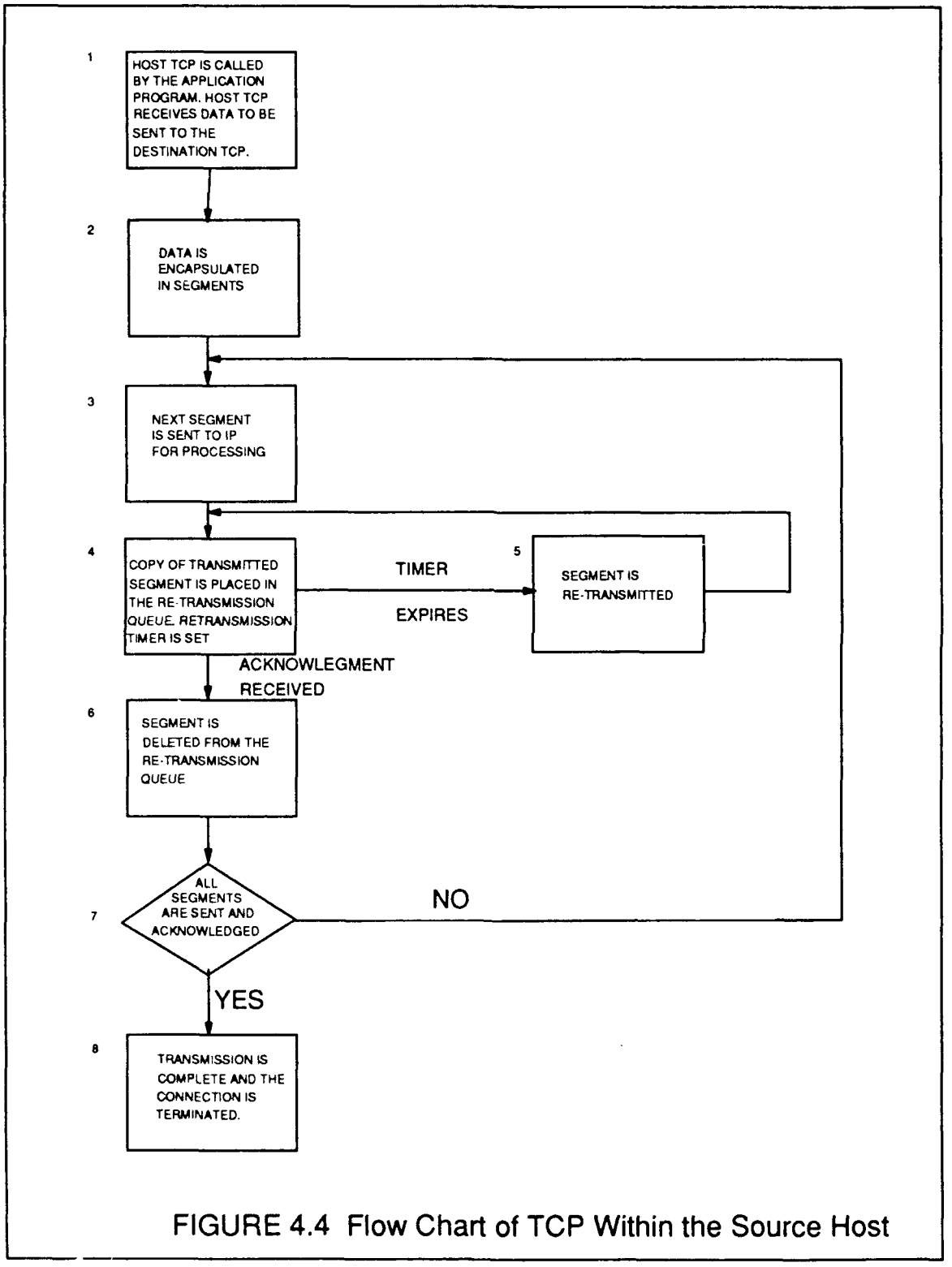


FIGURE 4.4 Flow Chart of TCP Within the Source Host

If all segments have been sent and acknowledged (7), the transmission is complete and the connection is terminated (8). TCP continues to send segments and await acknowledgments until all segments are acknowledged or until the connection has been terminated.

Data must be delivered and acknowledged in a timely manner or the TCP will retransmit the segment. Increased system congestion will result if the retransmission timer is too small. Long delays will be encountered if the retransmission timer is too large. To accommodate various delays in transmission across the interconnected networks, TCP uses an adaptive retransmission algorithm that monitors delays and adjusts timeouts accordingly.

TCP maintains the reliability necessary to give credibility to TCP/IP. The use of sequence numbers and end to end acknowledgments are essential in maintaining reliable communications.

E. FLOW CONTROL

Flow control is required within interconnected networks because of the various processing speeds and capabilities associated with each network. Without flow control, data may arrive faster than it can be processed. Any additional segments received when the destination buffer is full will be discarded and must be retransmitted from the source. This retransmission

results in inefficiencies due to an increase in network congestion and a decrease in system performance.

In TCP, flow control is achieved by using the variable sliding window method. The window is a field in the control information of the TCP segment that can vary in value from zero to 65,535 bytes.

The value of the window field is established by the destination TCP. The window indicates the maximum number of bytes that the sender is permitted to transmit until a new segment arrives from the destination. Basically, the window approximates the buffer size which the destination currently has for additional data.

Flow control within a connection can be controlled by this window value. To increase the rate of data flow, the window size is increased. A reduction in window size results in a decrease of the data flow.

Decreasing the size of the window can lead to problems if the acknowledgment segments arrive out of order. Figure 4.5 [Ref 11] illustrates a potential problem. After the source TCP has sent segment one, the destination TCP responds with an acknowledgment of two and a window availability of eight. A reallocation of resources may result in a reduction of the window size. In this example, it is reduced to three.

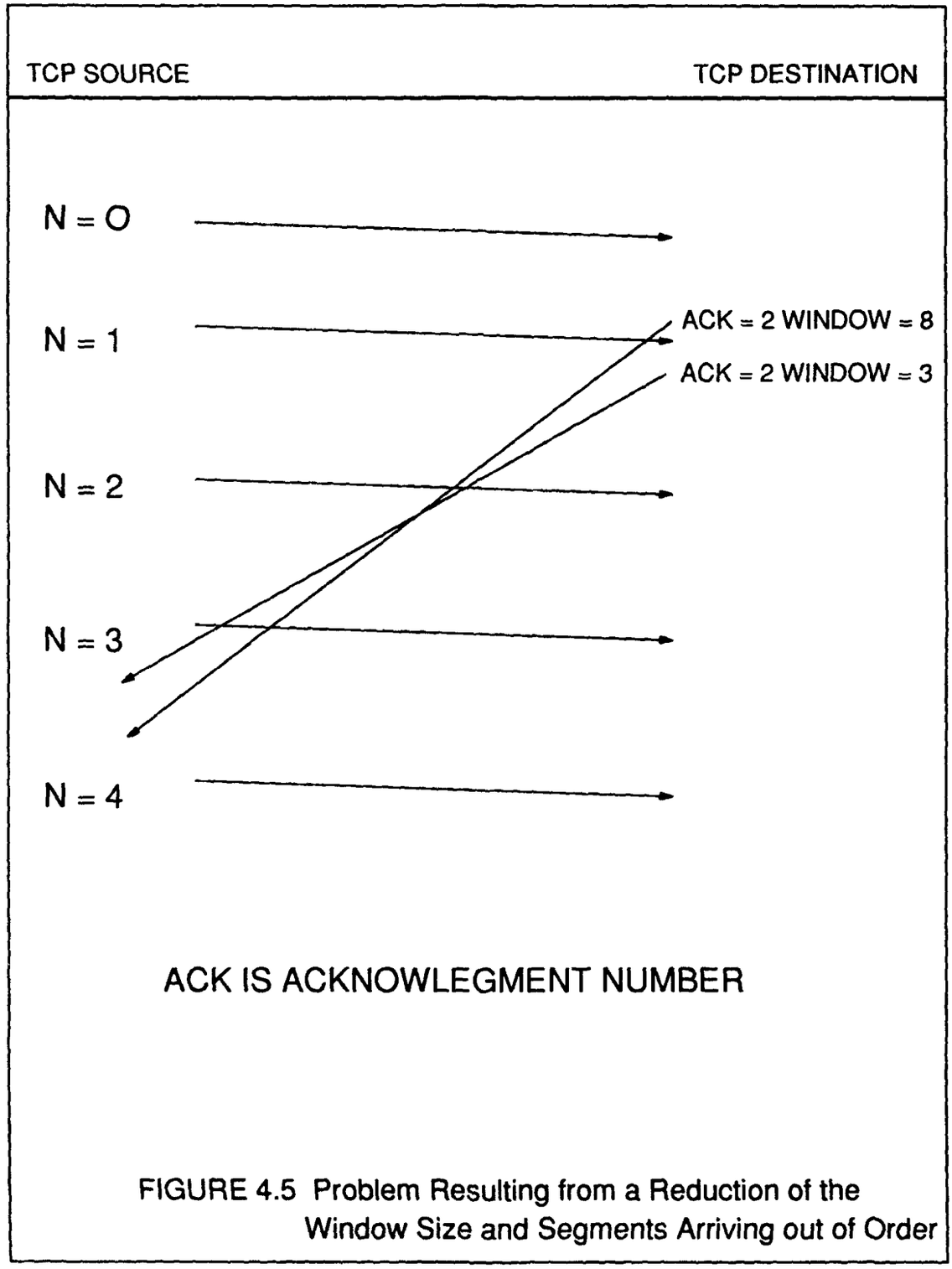
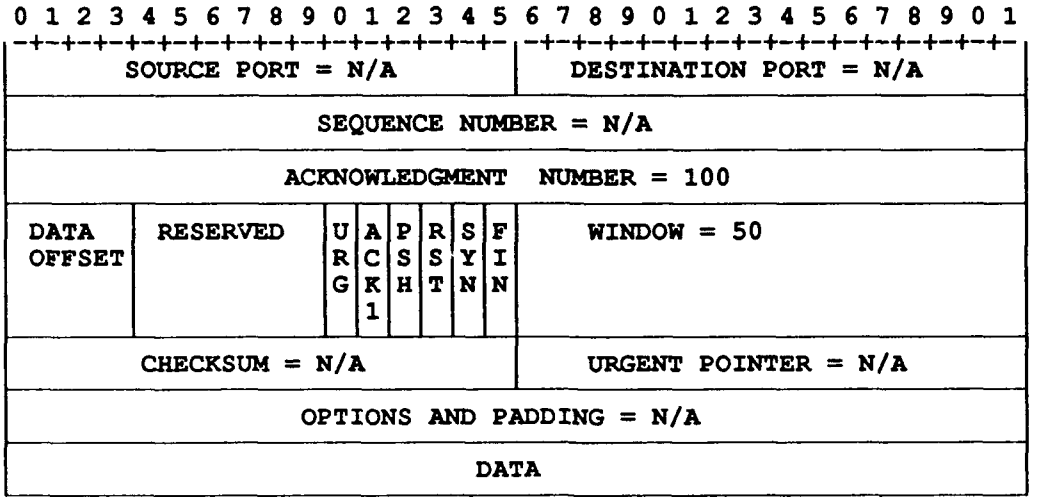


FIGURE 4.5 Problem Resulting from a Reduction of the Window Size and Segments Arriving out of Order

If this new allocation arrives at the source prior to the first acknowledgment, it would appear to the source that the destination has granted an allocation of three and then obtained additional resources and can now handle eight bytes. In actuality, the destination can only handle three additional bytes. Inefficiencies will result because the additional bytes sent by the source cannot be processed by the destination and must be discarded and retransmitted by the source.

The flow control window is used in conjunction with the acknowledgment number to give an allowance to the sending TCP on the permissible sequence numbers that may be sent. The maximum permissible sequence number can be computed by the formula ((acknowledgment number + window) - 1). Figure 4.6 illustrates this computation.

Given: Acknowledgment segment
Acknowledgment number: 100
Window: 50



This acknowledgment from the destination TCP indicates that all sequence numbers through 100 - 1 have been received and the next expected sequence number is 100. Permissible sequence numbers are from 100 to ((100 + 50) - 1) = 149.

Figure 4.6 Computation of Permissible Sequence Numbers that can be sent by the Source TCP.

Flow control is essential to system performance. At any given time, the source TCP knows exactly the amount of sequence numbers that it is permitted to send.

F. SUMMARY

The first transport layer protocol of the military internetwork was designed to work in a perfect network. It simply passed data units to the network layer and assumed that all the data would arrive correctly to its destination. As the internetwork expanded in size, many different networks

with various reliabilities were added. TCP provides the reliability and flow control necessary to make TCP/IP viable for network communications.

TCP provides for the recovery and retransmission of lost, garbled, and out of sequence segments of data. It is capable of operating in a larger variety of systems.

Prior to transmitting data, both the TCP of the source host and the TCP of the destination host must establish a connection. This guarantees that both the source and destination are ready to send and receive data and also provides for the synchronization of control information.

Each byte of data sent by the source TCP has a sequence number associated with it. Subsequently, each byte of data received by TCP is acknowledged by the same sequence number. If a segment is lost, the source will retransmit the segment until an acknowledgment is received. Sequence numbers also allow the destination to place the segments in their proper order.

Flow control within the DDN is imperative because of various network speeds and capabilities. Without the proper use of flow control, data may arrive faster than it can be processed. Flow control is achieved within TCP by using the variable size sliding window method when transferring data.

TCP is an effective protocol. It ensures the integrity of the transferred data. However, its abilities cannot be realized without the use of the unreliable IP.

with various reliabilities were added. TCP provides the reliability and flow control necessary to make TCP/IP viable for network communications.

TCP provides for the recovery and retransmission of lost, garbled, and out of sequence segments of data. It is capable of operating in a larger variety of systems.

Prior to transmitting data, both the TCP of the source host and the TCP of the destination host must establish a connection. This guarantees that both the source and destination are ready to send and receive data and also provides for the synchronization of control information.

Each byte of data sent by the source TCP has a sequence number associated with it. Subsequently, each byte of data received by TCP is acknowledged by the same sequence number. If a segment is lost, the source will retransmit the segment until an acknowledgment is received. Sequence numbers also allow the destination to place the segments in their proper order.

Flow control within the DDN is imperative because of various network speeds and capabilities. Without the proper use of flow control, data may arrive faster than it can be processed. Flow control is achieved within TCP by using the variable size sliding window method when transferring data.

TCP is an effective protocol. It ensures the integrity of the transferred data. However, its abilities cannot be realized without the use of the unreliable IP.

There are some inefficiencies that are related to TCP. The inefficiencies result from the fact that at least 160 bits of information are required in the segment header for each segment of data transmitted. On a typical segment, i.e. one that does not contain urgent data, the following fields of the segment header contain irrelevant information:

Segment Field Name	Size
Data Offset	4
Reserved	6
Flags	6
Urgent Pointer	16

Thus, 20% of the bits in a typical segment header are transmitted across the network but are not utilized by the source or destination TCP. In fact, an inefficiency is realized in the transmission of every TCP segment because there is not a single TCP segment that will make applicable use of each of the fields in a segment header.

These inefficiencies exist because TCP utilizes a standard header to differentiate various types of TCP segments. Unless TCP drastically changes its procedures to distinguish the different types of TCP segments, the inefficiency noted above cannot be avoided.

V. CONCLUSIONS AND RECOMMENDATIONS

One of the advances of the 20th century has been the development of the computer network. Just a few years ago, computer networks were exotic tools used by only a few specialists. Today, the multitude of computers in use by the military are more likely to be part of an interconnected network than not. Utilizing the network has been simplified and many individuals with various backgrounds are able to use the networks for their daily activities.

A great deal of credit for simplifying the use of the military's interconnected networks can be given to the combined efforts of both the military and civilian groups who refined the capabilities of TCP/IP. By using TCP/IP, the majority of technical issues of interconnecting various computer technologies have become transparent to the user.

Simplified standard procedures for FTP, SMTP, and Telnet, make the military's interconnected networks accessible to users with little or no network communication experience. Thousands of users at a multitude of locations throughout the world unknowingly depend upon TCP/IP while utilizing the military's interconnected network in their daily activities.

Most military networks are designed to accomplish the goals of an individual group. The government's competitive procurement process awards

the purchase of the desired network to the lowest bidder who can accomplish the stated objectives at the individual installation. This has resulted in a variety of seemingly incompatible network technologies from a multitude of vendors. TCP/IP was developed to hide the technical differences between networks and make communication between the networks independent of all underlying hardware technology.

The development of the military's interconnected network and its use of TCP/IP has been very successful and widely accepted throughout the military. Clearly, its abilities and its ease of use have enhanced the functional effectiveness of the users.

Despite TCP/IP's success, there is room for some improvements. One problem with TCP/IP is that the international OSI communication protocols are dominating today's marketplace. In order for TCP and IP to survive in the future, they must modernize and evolve to become completely compatible with the OSI protocols.

This thesis explored many aspects of the TCP/IP technology. Based upon descriptions provided, flowcharts detailing the series of procedures of numerous functions of both IP and TCP were created. Additionally, inefficient TCP/IP functions were noted in the transmission of irrelevant data bits in the header segment of every TCP segment and in the IP fragmentation procedure.

In order to eliminate the inefficiencies of sending irrelevant data bits in each TCP segment, TCP might consider evolving to similar techniques used in the OSI transport layer. The OSI transport layer has avoided the problem of sending irrelevant data bits by supporting nine different types of transport segments.

Segments containing synchronization data, acknowledgments, urgent data, etc., are immediately recognized within the OSI transport layer without the transmission of irrelevant bits of data. By contrast, TCP utilizes one standard header and flags within the header to identify the type of segment transmitted. A transmitted TCP segment may contain a segment header with up to 20% of control information that does not pertain to the function of the segment.

An immediate increase in efficiency can be realized by slightly modifying the IP fragmentation procedure as discussed in Chapter III.D.4. The basic principle behind this modification is that fragmented IP datagrams are filled to their maximum capacity prior to being transmitted over the network layer. In the current algorithm, fragmented datagrams are continually divided in half until they can be processed by the network layer. This may produce partially filled datagrams. IP exerts the same effort in processing a partially full datagram and a completely full datagram because only the datagram headers

are processed. Therefore, in order to reduce congestion by limiting the number of active datagrams, it is logical to send full datagrams whenever it is possible.

TCP's evolution to distinctive headers for each type of segment and the suggested modification of the IP fragmentation procedure would slightly increase the total system's efficiency. However, such an approach would be a complex operation involving a major programming effort. These recommendations would only be feasible if the congestion of data packets became a prohibitive factor in daily network operations and no other congestion reduction solutions prevalent.

Overall, TCP/IP is quite clever in its procedures of dealing with a multitude of seemingly incompatible network technologies. Its success in transmitting reliable data and its acceptance in the military cannot be disputed. However, as the military's interconnected network expands, congestion of data packets will become an ever increasing problem. Adoption of the suggestions listed above will certainly alleviate a small percentage of the congestion problem. Advances in hardware technology to facilitate the expeditious handling of data packets within the network may possibly make the problem of congestion a problem of the past.

LIST OF REFERENCES

1. Comer, D., *Internetworking with TCP/IP*, p.5, Prentice Hall, Inc., 1988.

2. Stallings, W., *Handbook of Computer Communication Standards Volume 3*, p.3, Macmillan Publishing Company, 1988.

3. Electronic mail dated 10 September 1990, from Lt. W. Hartung, USN, of the Defense Communication Agency in Washington, D.C.

4. Stallings, W., *Data and Computer Communication*, p.405, Macmillan Publishing Company, 1988.

5. Stallings, W., *Handbook of Computer Communication Standards Volume 3*, p.8, Macmillan Publishing Company, 1988.

6. Comer, D., *Internetworking with TCP/IP*, p.40, Prentice Hall, Inc., 1988.

7. Postel, J. (ed), "Internet Protocol - DARPA Internet Program Protocol Specification", *RFC 791*, p. 11, USC/Information Science Institute, September 1981.

8. Postel, J. (ed), "Internet Protocol - DARPA Internet Program Protocol Specification", *RFC 791*, p. 28, USC/Information Science Institute, September 1981.

9. Postel, J. (ed), "Transmission Control Protocol - DARPA Internet Program Protocol Specification", *RFC 793*, p.15, USC/Information Science Institute, September 1981.

10. Tannenbaum, A.S., *Computer Networks*, p.396, Prentice Hall, Inc., 1989.

11. Stallings, W., *Data and Computer Communication*, p. 494, Macmillan Publishing Company, 1988.

12. Quarterman, J.S., and Josiah C. Hoskins, "Notable Computer Networks", *Communications of the ACM Volume 29*, Number 10, pp. 932-971, October 1986.

APPENDIX A

List of Acronyms

ACK	Acknowledgment
ARPANET	Advance Research Projects Agency Network
ASCII	American Standard Code for Information Interchange
DARPA	Defense Advance Research Projects Agency
DCA	Defense Communication Agency
FO	Fragment Offset
FTP	File Transfer Protocol
ICMP	Internet Control Message Processor
IHL	Internet Header Length
IP	Internet Protocol
ISN	Initial Sequence Number
MF	More Flag
MILNET	Military Network
MTU	Maximum Transmission Unit
NID	Network Identification
NPS	Naval Postgraduate School, Monterey, California
OSI	Open System Interconnection
PSN	Packet Switch Node

SMTP	Simple Mail Transfer Protocol
SUBNET	Subnetwork
TAC	Terminal Access Controller
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TDL	Total Data Length
TL	Total Length
TTL	Time to Live

APPENDIX B

A partial list of many wide area networks accessible by the military internetwork is provided below [Ref 11]. To be part of the military internetwork, a network must utilize TCP/IP and be connected (either directly connected or indirectly connected) to a network that is part of the military internetwork.

There are a multitude of small local area networks that are indirectly attached to the military's internetwork. These subnets, too numerous to mention, are also considered part of the military internetwork.

The military's internetwork is also known as the Defense Data Network (DDN). The DDN is mostly confined to the United States but there are links to Germany, Japan, Korea, Australia, Israel, France, Sweden, and the United Kingdom.

Originally, the ARPANET and the MILNET were the main networks of the DDN. In April 1990, the ARPANET and MILNET were reorganized into groups of easily managed interconnected networks. The new domains of this reorganization are listed below:

COM - Commercial Organizations
EDU - Educational Organizations
GOV - Civilian Government Organizations
MIL - Department of Defense Organizations
NET - Administrative Organization of Networks
ORG - Other Organizations

Other networks are also connected to the DDN. The following is a list of classified networks that are connected:

DISNET - Defense Investigative Network
SCINET - Sensitive Compartmented Information Network
WINGS - WWMCCS Intercomputer Network Communication System (WWMCCS stands for World Wide Military Command and Control System).

Additional wide area networks accessible by the DDN include:

BITNET - (Because It's Time Network) is a cooperative network that forms a communication link between universities and research centers. BITNET also includes the European network EARN and the Canadian Network NETNORTH.

CDNNET - Connects research and educational communities in Canada

CSNET - Facilitates research and development in computer science by providing electronic mail service to individuals with access to the DDN to individuals without access to the DDN. CSNET includes PHONENET, CYPRESS, and X25NET

DFN - (Deutsche Forshungnetz) the national research network of Germany.

DRENET - Canada's military network

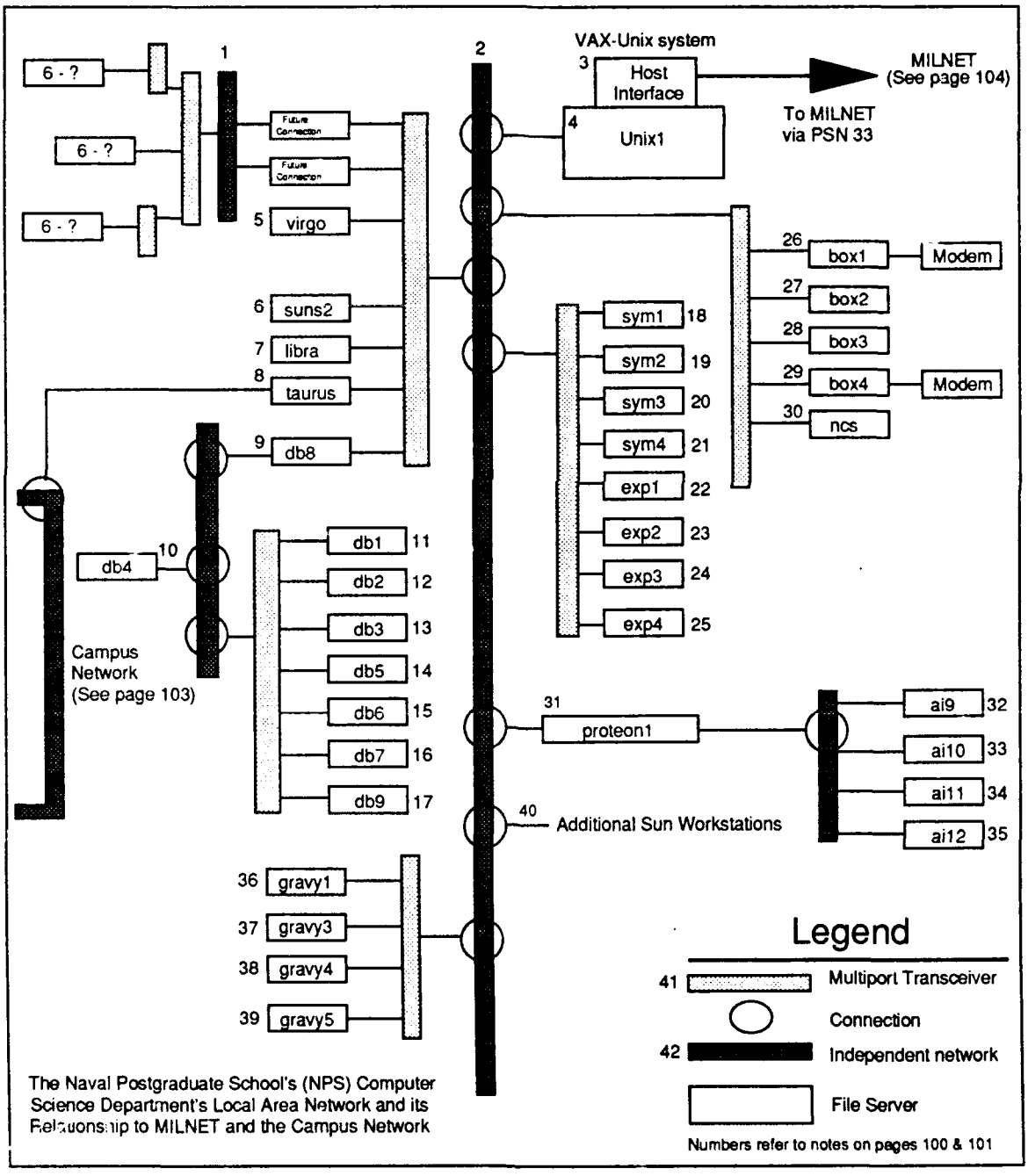
JANET - Connects large university computer centers and research facilities in the United Kingdom

MFENET - Magnetic Fussion Energy Network located at Lawrence Livermore National Laboratories

SATNET - A satellite linked network

SPAN - Space Physics Analysis Network used by National Aeronautics and Space Administration (NASA)

APPENDIX C


6 - ?
6 - ?
6 - ?
1
Future Connection
Future Connection
5 virgo
6 suns2
7 libra
8 taurus
9 db8
2
VAX-Unix system
3 Host Interface
4 Unix1
MILNET
(See page 104)
To MILNET
via PSN 33
26 box1
Modem
27 box2
28 box3
29 box4
Modem
30 ncs
sym1 18
sym2 19
sym3 20
sym4 21
exp1 22
exp2 23
exp3 24
exp4 25
db4 10
db1 11
db2 12
db3 13
db5 14
db6 15
db7 16
db9 17
Campus
Network
(See page 103)
31 proteon1
ai9 32
ai10 33
ai11 34
ai12 35
40 Additional Sun Workstations
36 gravy1
37 gravy3
38 gravy4
39 gravy5
Legend
41 Multiport Transceiver
Connection
42 Independent network
File Server
Numbers refer to notes on pages 100 & 101
The Naval Postgraduate School's (NPS) Computer Science Department's Local Area Network and its Relationship to MILNET and the Campus Network

NOTES EXPLAINING THE COMPUTER SCIENCE DEPARTMENT'S LOCAL AREA NETWORK

Reference Number	Comment
1.	a future ethernet network
2.	Computer Science Department's backbone ethernet cable
3.	interface model DEC 3100
4.	VAX 11/785 Minicomputer
5.	a Sun Workstation
6.	a Sun Workstation
7.	a Sun Workstation
8.	a Sun Workstation, department server, and link to the campus network
9.	an ISIV16 Workstation UNIX
10.	an ISIV16 Workstation UNIX
11 - 17.	ISIV16 Workstations UNIX that comprise the database lab
18.	Symbolics 3675 LISP
19.	Symbolics 3640 LISP
20.	Symbolics 3640 LISP
21.	Symbolics 3650 LISP
22 -25.	TI Explorer LISP (18-25 are used for LISP operations)

26 - 30. Bridge Terminal Servers (8 line modems are attached to box 1 and box 4)

31. a dedicated router

32 - 35. Sun Workstations that comprise the artificial intelligence lab

36 - 39. Iris Graphics Workstations that comprise the graphics lab

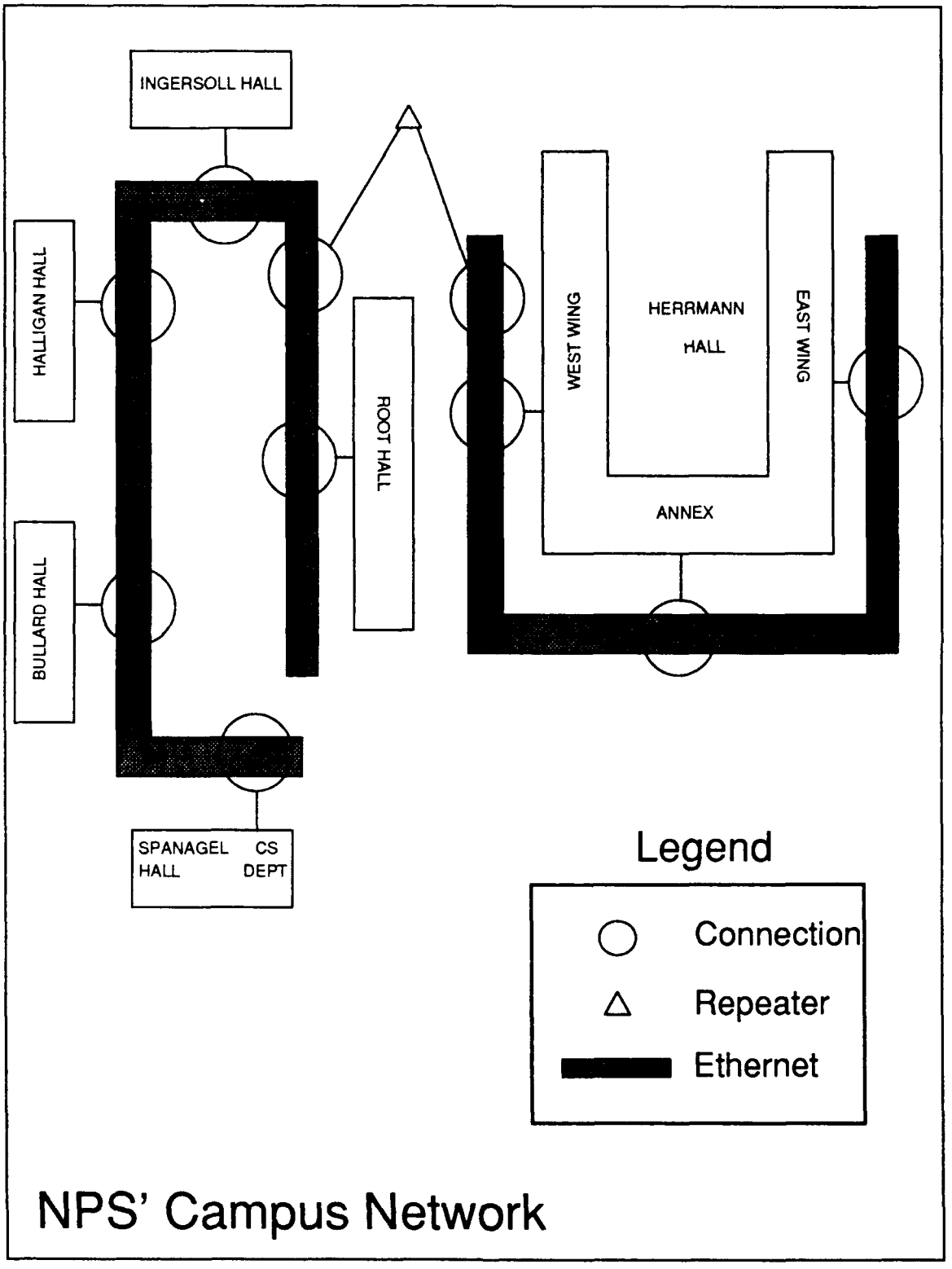
40. a remaining 39 Sun Workstations have access to the ethernet (a total list of all 47 Sun Workstations is provided on page 102)

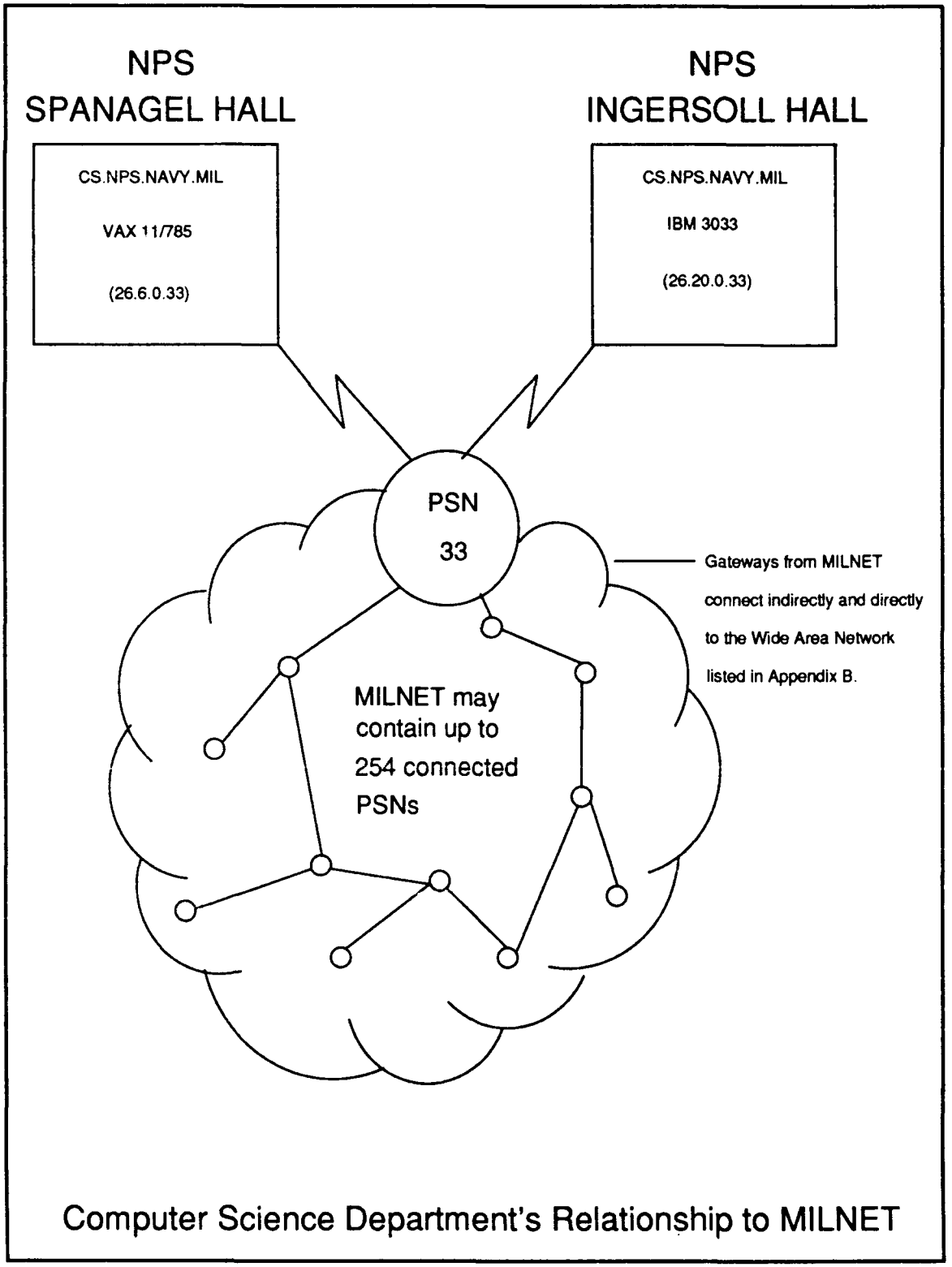
41. Multiport transceivers allow 8 individual file servers to share one common connection to the ethernet

42. ethernet network using carrier sense multiple access with collision detection (CSMA/CD)

The following is a list of the 47 Sun Workstations as of 4 Dec 90 attached to the Computer Science Department's network:

ai9
ai10
ai11
ai12
aldebaran
cephus
europa
io
libra
lyra
maroon
orion
sun1
sun10
sun11
sun12
sun14
sun16
sun17
sun18
sun19
sun20
sun21
sun22
sun23
sun24
sun30
sun31
sun32
sun33
sun34
sun35
sun40
sun41
sun42
sun43
sun44
sun45
sun46
sun50
sundb1
sundb2
suns2
suns5
taurus
voyager
virgo


INGERSOLL HALL
HALLIGAN HALL
BULLARD HALL
ROOT HALL
WEST WING
HERRMANN HALL
EAST WING
ANNEX
SPANAGEL HALL
CS DEPT
Legend
Connection
Repeater
Ethernet
NPS' Campus Network



Computer Science Department's Relationship to MILNET

INITIAL DISTRIBUTION LIST

1.	Professor G.M. Lundy Code CS/LN Computer Science Department Naval Postgraduate School Monterey, California 93943-5100	5
2.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
3.	Dudley Knox Library Code 52 Naval Postgraduate School Monterey, California 93943-5002	2
4.	Captain Bruce R. Eikenberg c/o Director Computer Sciences School Headquarters and Service Battalion Marine Air-Ground Training and Education Center, MCCDC Quantico, Virginia 22134-5050	2
5.	Commandant of the Marine Corps Code TE06 Headquarters, U.S. Marine Corps Washington, D.C. 20380-0001	1
6.	Professor M.T. Shing Code CS/SH Computer Science Department Naval Postgraduate School Monterey, California 93943-5100	1